Itala Answer Key - Intro to Applied linear algebra textbook solutions

Linear Algebra For Engineers (Cornell University)

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

July 14, 2018

CAMBRIDGE
UNIVERSITY PRESS

For

Anna, Nicholas, and Nora

Daniël and Margriet

Contents

Chapter 1

Vectors

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

Solution.

(a) *This equation is valid notation and is true.* Both sides of the equation are 3-vectors with equal entries.

(b) *This equation doesn't make sense.* The expression $[\ 1, \quad 2, \quad 1\ ]$ is not valid notation.

(c) *This equation is valid notation and is true.* Both sides of the equation use stacked vector notation and are equivalent to the vector $(1, 2, 1)$.

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) $[\ a\ \ b\ ] + 4c$.

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

Solution.

(a) *Correct.* Each of the vectors has length 10, so you can add them.

(b) *Correct.* The length is the sum of the lengths of parts that are stacked, $10+10+11 = 31$.

(c) *Incorrect.* $2a$ has length 10, but c has length 20, so you cannot add them.

(d) *Correct.* The size of both vectors is 11.

(e) *Correct.* (a, b) is a stacked vector of size 20, and $((a, b), a)$ is a stacked vector of size 30.

(f) *Incorrect.* $[\ a\ \ b\ ]$ is not a vector (it is a 10×2 matrix, which we study later), while $4c$ is a 20-vector.

(g) *Correct.* Stacking a and b gives a vector of length 20 which can be added to c.

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) $a = 0 = b$.

Solution.

(a) Correct and unambiguous. 0 is the zero vector of length 10.

(b) Incorrect. The left-hand side has length 10. The length of the right-hand is at least 21.

(c) Correct but ambiguous. It is impossible to determine the sizes of the zero vectors from the equation.

(d) Incorrect. The first equality only makes sense if 0 is the zero vector of length 10. The second equality only makes sense if 0 is the zero vector of length 20.

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.

Solution.

(a) $w = (d, d, d, d, d, d, d)$.

(b) $d = w_{1:24}$. There are other solutions, for example, $d = w_{25:48}$.

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.

Solution.

(a) On most days the business neither receives nor makes cash payments.

(b) The customer has purchased only a few of the n products.

(c) The portfolio invests (long or short) in only a small number of stocks.

(d) The project requires only a few types of materials.

(e) Most pixels have a brightness value of zero, so the image consists of mostly black space. This is the case, for example, in astronomy, where many imaging methods leverage the assumption that there are a few light sources against a dark sky.

(f) On most days it didn't rain.

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.

Solution. $d = x_{2:n} - x_{1:n-1}$.

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.

Solution. We have $y = 2x - \mathbf{1}$. To see this, we note that $y_i = 2x_i - 1$. When $x_i = 1$, we have $y_1 = 2 \cdot 1 - 1 = +1$; when $x_i = 0$, we have $y_1 = 2 \cdot 0 - 1 = -1$.

Conversely, we have $x = (1/2)(y + \mathbf{1})$.

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.

Solution. The profit for item i is $p_i s_i$, so the total profit is $\sum_i p_i s_i = p^T s$. In other words, the total profit is just the inner product of p and s.

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.

Solution.

(a) The total number of symptoms the patient has is $\mathbf{1}^T s$.

(b) The patient exhibits five out of the first ten symptoms can be expressed as $a^T s = 5$, where the 20-vector a is given by $a = (\mathbf{1}_{10}, 0_{10})$. (The subscripts give the dimensions.)

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.

Solution. To convert the total homework score to a scale of 0–100, we add up the raw scores and multiply by $100/80 = 1.25$, $1.25(r_1 + \cdots + r_8)$. The midterm score on a 0–100 scale is $(100/120)r_9 = 0.8333 r_9$, and the final exam score on a 0–100 scale is $(100/160)r_{10} = 0.625 r_{10}$. We multiply these by the weights 25%, 35%, and 40% to get the total score,

$$\begin{aligned} s &= (0.25)(1.25)(r_1 + \cdots + r_8) + (0.35)(0.8) r_9 + (0.45)(0.625) r_{10} \\ &= (0.3125\ \mathbf{1}_8, 0.2917, 0.25)^T r. \end{aligned}$$

So, $w = (0.3125\ \mathbf{1}_8, 0.2917, 0.2500)$, where the subscript 8 on the ones vector tells us that it has size 8.

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.)

Solution.

(a) $\mathbf{1}^T w$ is total number of words in the document.

(b) $w_{282} = 0$ means that word 282 does not appear in the document.

(c) $h = w/(\mathbf{1}^T w)$. We simply divide the word count vector by the total number of words in the document.

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.

Solution. The total value held in currency i, in USD, is given by $a_i c_i$, where a_i is the USD value of one unit of currency i. (So $a_i = 1$.) The total USD value is then $a_1 c_1 + \cdots + a_5 c_5 = a^T c$, the inner product of the exchange rate vector a and the currency holdings vector c.

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.)

Solution.

(a) The total population is $\mathbf{1}^T x$.

(b) The total number of people aged 65 or over is given by $a^T x$, where $a = (0_{65}, \mathbf{1}_{35})$. (The subscripts give the dimensions of the zero and ones vectors.)

(c) The sum of the ages across the population is $(0, 1, 2, \ldots, 99)^T x$. And so the average age is given by

$$\frac{(0, 1, 2, \ldots, 99)^T x}{\mathbf{1}^T x}.$$

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.

Solution. If h is neutral to a sector represented by f, we have $f^T h = 0$. But this means

$$f_1 h_1 + \cdots + f_n h_n = 0.$$

Each term in this sum is the product of two nonnegative numbers, and is nonnegative. It follows that each term must be zero, *i.e.*, $f_i h_i = 0$ for $i = 1, \ldots, n$. This in turn means that when asset i is in the sector, so $f_i = 1$, we must have $h_i = 0$. In other words: A long only portfolio is neutral to a sector only if it does not invest in any assets in that sector.

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.

Solution. If we place the order with supplier i, the total price of the order is given by the inner product $p_i^T q$. We find the cheapest supplier by calculating $p_i^T q$ for $i = 1, \ldots, K$, and finding the minimum of these K numbers.

Suppose the cheapest and second cheapest supplier are suppliers i and j. By splitting the order in two, the total price is $(p_i^T q + p_j^T q)/2$. This is never less than when we place the order with one supplier. However, ordering from more than one supplier can be advantageous for other reasons than cost.

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.

Solution.

(a) Let x and y be nonnegative n-vectors. The inner product

$$x^T y = x_1 y_1 + x_2 y_2 + \cdots + x_n y_n$$

is nonnegative because each term in the sum is nonnegative.

(b) For each k, $x_k = 0$ or $y_k = 0$ (or both). Therefore only the following three combinations of zero-positive patterns are positive (here + stands for a positive entry):

x_k	y_k
+	0
0	+
0	0

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.

Solution. We are asked to write the T-vector

$$c = (1, 0, \ldots, 0, -(1+r)^{T-1})$$

as a linear combination of the $T-1$ vectors

$$l_t = (0, \ldots, 0, 1, -(1+r), 0, \ldots, 0), \quad t = 1, \ldots, T-1.$$

In the definition of l_t there are $t-1$ leading and $T-t-1$ trailing zeros, *i.e.*, the element 1 is in position t. There is only one way to do this:

$$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ -(1+r)^{T-1} \end{bmatrix} = \begin{bmatrix} 1 \\ -(1+r) \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} + (1+r) \begin{bmatrix} 0 \\ 1 \\ -(1+r) \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} + (1+r)^2 \begin{bmatrix} 0 \\ 0 \\ 1 \\ -(1+r) \\ \vdots \\ 0 \\ 0 \end{bmatrix} + \cdots + (1+r)^{T-2} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 1 \\ -(1+r) \end{bmatrix}.$$

In other words,

$$c = l_1 + (1+r)l_2 + (1+r)^2 l_3 + \cdots + (1+r)^{T-2} l_{T-1}.$$

The coefficients in the linear combination are 1, $1+r$, $(1+r)^2$, ..., $(1+r)^{T-1}$.

The idea is that you extend the length of an initial loan by taking out a new loan each period to cover the amount that you owe. So after taking out a loan for \$1 in period 1, you take out a loan for $\$(1+r)$ in period 2, and end up owing $\$(1+r)^2$ in period 3. Then you take out a loan for $\$(1+r)^2$ in period 3, and end up owing $\$(1+r)^3$ in period 4, and so on.

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.)

Solution. We will consider the case $m = k = 2$. We have

$$b_1 = \beta_1 a_1 + \beta_2 a_2, \qquad b_2 = \beta_3 a_1 + \beta_4 a_2.$$

Now assume that $c = \alpha_1 b_1 + \alpha_2 b_2$. Then we have

$$\begin{aligned} c &= \alpha_1 b_1 + \alpha_2 b_2 \\ &= \alpha_1(\beta_1 a_1 + \beta_2 a_2) + \alpha_2(\beta_3 a_1 + \beta_4 a_2) \\ &= (\alpha_1\beta_1 + \alpha_2\beta_3)a_1 + (\alpha_1\beta_2 + \alpha_2\beta_4)a_2. \end{aligned}$$

This shows that c is a linear combination of a_1 and a_1.

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

Solution.

(a) The prediction is $\hat{z}_{t+1} \approx z_t$. This means that the prediction of tomorrow's value is, simply, today's value.

(b) The prediction is $\hat{z}_{t+1} \approx z_t + (z_t - z_{t-1})$. In other words, the prediction is found by linearly extrapolating from yesterday's and today's values.

(c) The prediction is $\hat{z}_{t+1} \approx z_{t-6}$, which is the value six days ago, which is the same as one week before tomorrow. For example, if today is Sunday we predict the value for Monday that is last Monday's value.

(d) The prediction is $\hat{z}_{t+1} \approx 0.5z_t + 0.5z_{t-1}$, the average of today's and yesterday's values.

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?

Solution.

(a) $8 \times 100 \times 10^5 = 8\,10^7$ bytes.

(b) $100 \times 10^5 + 99 \times 10^5 \approx 2\,10^7$ flops.

(c) About 20 milliseconds.

Chapter 2

Linear functions

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}.)

Solution.

(a) *Not linear.* Choose $x = (1, 0)$, $y = (0, 1)$, $\alpha = \beta = 1/2$. Then $\alpha x + \beta y = (1/2, 1/2)$, $f(x) = f(y) = 1$, and

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

(b) *Linear.* The function can be written as an inner product $f(x) = a^T x$ for

$$a = (-1, 0, \ldots, 0, 1).$$

(c) *Not linear.* Choose $x = (-1, 0, 2)$, $y = (2, -1, 0)$, $\alpha = \beta = 1$. Then $\alpha x + \beta y = (1, -1, 2)$, $f(x) = f(y) = 0$, and

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

(d) *Linear.* If the size of the vector is even ($n = 2m$), there are m even and m odd indices, and $f(x) = a^T x$ with

$$a = (1/m, -1/m, 1/m, -1/m, \ldots, 1/m, -1/m).$$

If the size of the vector is odd ($n = 2m+1$), there are m even and $m+1$ odd indices, and $f(x) = a^T x$ with

$$a = (1/(m+1), -1/m, 1/(m+1), -1/m, \ldots, 1/(m+1)).$$

(e) *Linear.* We have $f(x) = a^T x$ for

$$a = (0, 0, \ldots, 0, -1, 2).$$

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 $T = a^T P + b$.

Solution. First we express T as $T = a^T P + b$. If we can find the coefficients $a = (a_1, a_2, a_3)$ and b, then we will be able to determine the temperature for any processor powers.

The four given data points give the following information:

$$\begin{aligned} 10a_1 + 10a_2 + 10a_3 + b &= 30 \\ 100a_1 + 10a_2 + 10a_3 + b &= 60 \\ 10a_1 + 100a_2 + 10a_3 + b &= 70 \\ 10a_1 + 10a_2 + 100a_3 + b &= 65. \end{aligned}$$

Subtracting the first equation from the second gives $90a_1 = 30$. Hence, $a_1 = 1/3$. Subtracting the first equation from the third gives $90a_2 = 40$, Hence $a_2 = 4/9$. Subtracting the first equation from the fourth gives $90a_3 = 35$, Hence $a_3 = 35/90$. Plugging in these values in any of the four equations and solving for b gives $b = 55/3$. So now we have a full model for the temperature for any processor powers:

$$T = (1/3)P_1 + (4/9)P_2 + (7/18)P_3 + 55/3.$$

We can now predict the temperature when all processor operate at power $P_i = P^{\text{same}}$:

$$T = (1/3 + 4/9 + 7/18)P^{\text{same}} + 55/3 = (7/6)P^{\text{same}} + 55/3.$$

This satisfies $T \leq 85$ provided $P^{\text{same}} \leq 57.1$.

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.

Solution.

$$s'(10) = s'(0) + \int_0^{10} F(u)\, du = s'(0) + f_1 + f_2 + \cdots + f_{10}.$$

This is an affine function of f: $s'(10) = a^T f + b$ with

$$a = \mathbf{1} = (1, 1, \ldots, 1), \qquad b = s'(0).$$

The velocity $s'(t)$ is a piecewise-linear function of t and therefore also easily integrated. This gives

$$s(10) = s(0) + \int_0^{10} s'(u)\, du = s(0) + 10s'(0) + \frac{19}{2} f_1 + \frac{17}{2} f_2 + \cdots + \frac{1}{2} f_{10}.$$

We see that $s(10)$ is also an affine function of f, and can be expressed as $s(10) = c^T f + d$

$$c = (19/2, 17/2, \ldots, 1/2), \qquad d = s(0) + 10s'(0).$$

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.

Solution. The correct answer is: *ϕ cannot be linear.* To see this, we note that the third point $(1, -1, -1)$ is the negative of the second point $(-1, 1, 1)$. If ϕ were linear, the two values of ϕ would need to be negatives of each other. But they are 1 and 1, not negatives of each other.

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.

Solution.

(a) We note that

$$\begin{bmatrix} 1 \\ -1 \end{bmatrix} = (1/2) \begin{bmatrix} 1 \\ 0 \end{bmatrix} + (1/2) \begin{bmatrix} 1 \\ -2 \end{bmatrix}.$$

This is a linear combination with coefficients that add up to one, so we must have

$$\psi(1, -1) = (1/2)\psi(1, 0) + (1/2)\psi(1, -2) = 3/2.$$

(b) The value of $\psi(2, -2)$ cannot be determined. For example, the two affine functions

$$\psi(x_1, x_2) = 1 - (1/2)x_2, \qquad \psi(x_1, x_2) = 2x_1 - (1/2)x_2 - 1$$

both satisfy $\psi(1, 0) = 1$ and $\psi(1, -2) = 2$, but have a different value at $(2, -2)$.

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).

Solution. We have $w = (\mathbf{1}_{15}, 2\mathbf{1}_{15})$ and $v = -45$.
The contribution to the score from questions 1–15 is given by $\mathbf{1}^T(a_{1:15} - \mathbf{1})$. The contribution to the scores from questions 16–30 is given by $2\mathbf{1}^T(a_{16:30} - \mathbf{1})$. So, we can write the overall score as

$$(\mathbf{1}_{15}, 2\mathbf{1}_{15})^T(a - \mathbf{1}) = (\mathbf{1}_{15}, 2\mathbf{1}_{15})^T a - 45.$$

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.
Solution. We will evaluate the right-hand side of the formula (2.4), and show it is the same as $f(x) = a^T x + b$. We first note that $f(0) = b$, and $f(e_i) = a_i + b$, so $f(e_i) - f(0) = a_i$. Then we have

$$f(0) + x_1\,(f(e_1) - f(0)) + \cdots + x_n\,(f(e_n) - f(0)) = b + a_1 x_1 + \cdots + a_n x_n = a^T x + b.$$

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.

Solution.

(a) The integral is

$$\int_\alpha^\beta p(x)\,dx = c_1(\beta - \alpha) + \frac{c_2}{2}(\beta^2 - \alpha^2) + \frac{c_3}{3}(\beta^3 - \alpha^3) + \cdots + \frac{c_n}{n}(\beta^n - \alpha^n).$$

Therefore

$$a = (\beta - \alpha, \frac{\beta^2 - \alpha^2}{2}, \frac{\beta^3 - \alpha^3}{3}, \ldots, \frac{\beta^n - \alpha^n}{n}).$$

(b) The derivative at $\hat{x}$ is

$$p'(\alpha) = c_2 + 2c_3\alpha + 3c_4\alpha^2 + \cdots + c_n(n-1)\alpha^{n-2}.$$

Therefore

$$b = (0, 1, 2\alpha, 3\alpha^2, \ldots, (n-1)\alpha^{n-2}).$$

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.
Solution.

(a) $f(z) = 1$ and $\nabla f(z) = (z_2, z_1) = (1, 1)$. Therefore

$$\hat{f}(x) = f(z) + \nabla f(z)^T(x - z) = 1 + \begin{bmatrix} 1 \\ 1 \end{bmatrix}^T \begin{bmatrix} x_1 - 1 \\ x_2 - 1 \end{bmatrix} = x_1 + x_2 - 1.$$

(b)

x	$f(x)$	$\hat{f}(x)$
$(1, 1)$	1	1
$(1.05, 0.95)$	0.9975	1
$(0.85, 1.25)$	1.0625	1.1
$(-1, 2)$	-2	0

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 $\hat{y}$.

Solution.

(a) *False.* $\hat{y}$ can be negative, for example, if v is negative and very large.

(b) *True.* Since $\hat{y} = \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_8 x_8 + v$, the only term that depends on x_2 is $\beta_2 x_2$, which is zero if $\beta_2 = 0$.

(c) *True.* The change in $\hat{y}$ when we change x_6 by δ, keeping all other entries of x the same, is $\beta_6\delta$.

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.

Solution. The prediction $\hat{y}$ only depends on a few of features.

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%.

Solution.

(a) The regression model predicts that the profit on product 3 decreases if its price is increased.

(b) We choose the product with the largest absolute value of the coefficient β_i. If $\beta_i > 0$, we take $x_i = 0.01$, *i.e.*, increase the price of product i to $p_i^{\text{new}} = 1.01p_i$. If $\beta_i < 0$, we take $x_i = -0.01$, *i.e.*, decrease the price to $p_i^{\text{new}} = 0.99p_i$. The predicted total profit is

$$\hat{P} = P + \beta_i x_i = P + |\beta_i||x_i| = P + 0.01|\beta_i|.$$

(c) Suppose β_i and β_j are the largest two coefficients in absolute value. We choose products i and j and adjust their prices as in part (b). The predicted total profit is

$$\hat{P} = P + 0.01(|\beta_i| + |\beta_j|).$$

Chapter 3

Norm and distance

Exercises

3.1 *Distance between Boolean vectors.* Suppose that x and y are Boolean n-vectors, which means that each of their entries is either 0 or 1. What is their distance $\|x - y\|$?

Solution. For each i, $x_i - y_i$ is either -1, 0, or 1; $(x_i - y_i)^2$ is zero (if $x_i = y_i$) or 1 (if $x_i \neq y_i$). Summing over i, we find that $\|x - y\|^2$ is the total number of entries in which x and y differ. The distance is then the squareroot of the number of entries in which x and y differ.

3.2 *RMS value and average of block vectors.* Let x be a block vector with two vector elements, $x = (a, b)$, where a and b are vectors of size n and m, respectively.

(a) Express $\mathbf{rms}(x)$ in terms of $\mathbf{rms}(a)$, $\mathbf{rms}(b)$, m, and n.

(b) Express $\mathbf{avg}(x)$ in terms of $\mathbf{avg}(a)$, $\mathbf{avg}(b)$, m, and n.

Solution.

(a) The RMS value of the vector $x = (a_1, \ldots, a_n, b_1, \ldots, b_m)$ is

$$\begin{aligned}
\mathbf{rms}(x) &= \left(\frac{a_1^2 + \cdots + a_n^2 + b_1^2 + \cdots + b_m^2}{n+m}\right)^{1/2} \\
&= \left(\frac{n(a_1^2 + \cdots + a_n^2)/n + m(b_1^2 + \cdots + b_m^2)/m}{n+m}\right)^{1/2} \\
&= \left(\frac{n\,\mathbf{rms}(a)^2 + m\,\mathbf{rms}(b)^2}{n+m}\right)^{1/2}.
\end{aligned}$$

(b) The average of the vector $x = (a_1, \ldots, a_n, b_1, \ldots, b_m)$ is

$$\mathbf{avg}(x) = \frac{a_1 + \cdots + a_n + b_1 + \cdots + b_m}{n+m} = \frac{n\,\mathbf{avg}(a) + m\,\mathbf{avg}(b)}{n+m}.$$

3.3 *Reverse triangle inequality.* Suppose a and b are vectors of the same size. The triangle inequality states that $\|a + b\| \leq \|a\| + \|b\|$. Show that we also have $\|a + b\| \geq \|a\| - \|b\|$. *Hints.* Draw a picture to get the idea. To show the inequality, apply the triangle inequality to $(a + b) + (-b)$.

Solution. Following the hint (always a good idea), we have

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

Since $(a + b) + (-b) = a$, we have $\|a\| \leq \|a + b\| + \|b\|$. Subtracting $\|b\|$ from both sides gives the inequality we want.

3.4 *Norm identities.* Verify that the following identities hold for any two vectors a and b of the same size.

(a) $(a + b)^T(a - b) = \|a\|^2 - \|b\|^2$.

(b) $\|a + b\|^2 + \|a - b\|^2 = 2(\|a\|^2 + \|b\|^2)$. This is called the *parallelogram law*.

Solution.

(a)

$$\begin{aligned}
(a + b)^T(a - b) &= a^T(a - b) + b^T(a - b) \\
&= a^Ta - a^Tb + b^Ta - b^Tb \\
&= a^Ta - a^Tb + a^Tb - b^Tb \\
&= a^Ta - b^Tb \\
&= \|a\|^2 - \|b\|^2.
\end{aligned}$$

(b)

$$\begin{aligned}\|a+b\|^2 + \|a-b\|^2 &= (a+b)^T(a+b) + (a-b)^T(a-b)\\ &= a^Ta + b^Ta + a^Tb + b^Tb + a^Ta - b^Ta - a^Tb + b^Tb\\ &= 2a^Ta + 2b^Tb\\ &= 2\|a\|^2 + 2\|b\|^2.\end{aligned}$$

3.5 *General norms.* Any real-valued function f that satisfies the four properties given on page 46 (nonnegative homogeneity, triangle inequality, nonnegativity, and definiteness) is called a *vector norm*, and is usually written as $f(x) = \|x\|_{\mathrm{mn}}$, where the subscript is some kind of identifier or mnemonic to identify it. The most commonly used norm is the one we use in this book, the Euclidean norm, which is sometimes written with the subscript 2, as $\|x\|_2$. Two other common vector norms for n-vectors are the 1-*norm* $\|x\|_1$ and the ∞-*norm* $\|x\|_\infty$, defined as

$$\|x\|_1 = |x_1| + \cdots + |x_n|, \qquad \|x\|_\infty = \max\{|x_1|, \ldots, |x_n|\}.$$

These norms are the sum and the maximum of the absolute values of the entries in the vector, respectively. The 1-norm and the ∞-norm arise in some recent and advanced applications, but we will not encounter them in this book.

Verify that the 1-norm and the ∞-norm satisfy the four norm properties listed on page 46.

Solution. First the 1-norm.

(a) *Homogeneity.*

$$\begin{aligned}\|\beta x\|_1 &= |\beta x_1| + |\beta x_2| + \cdots + |\beta x_n|\\ &= |\beta|\,|x_1| + |\beta|\,|x_2| + \cdots + |\beta|\,|x_n|\\ &= |\beta|\,(|x_1| + |x_2| + \cdots + |x_n|)\\ &= |\beta|\,\|x\|_1.\end{aligned}$$

(b) *Triangle inequality.*

$$\begin{aligned}\|x+y\|_1 &= |x_1+y_1| + |x_2+y_2| + \cdots + |x_n+y_n|\\ &\leq |x_1| + |y_1| + |x_2| + |y_2| + \cdots + |x_n| + |y_n|\\ &= \|x\|_1 + \|y\|_1.\end{aligned}$$

(c) *Nonnegativity.* Each term in $\|x\|_1 = |x_1| + \cdots + |x_n|$ is nonnegative.

(d) *Definiteness.* $\|x\|_1 = 0$ only if $|x_1| = \cdots = |x_n| = 0$.

Next the ∞-norm.

(a) *Homogeneity.*

$$\begin{aligned}\|\beta x\|_\infty &= \max\{|\beta x_1|, |\beta x_2|, \ldots, |\beta x_n|\}\\ &= \max\{|\beta|\,|x_1|, |\beta|\,|x_2|, \ldots, |\beta|\,|x_n|\}\\ &= |\beta| \max\{|x_1|, \ldots, |x_n|\}\\ &= |\beta|\,\|x\|_\infty.\end{aligned}$$

(b) *Triangle inequality.*

$$\begin{aligned}\|x+y\|_\infty &= \max\{|x_1+y_1|, |x_2+y_2|, \ldots, |x_n+y_n|\}\\ &\leq \max\{|x_1|+|y_1|, |x_2|+|y_2|, \ldots, |x_n|+|y_n|\}\\ &\leq \max\{|x_1|, |x_2|, \ldots, |x_n|\} + \max\{|y_1|, |y_2|, \ldots, |y_n|\}\\ &= \|x\|_\infty + \|y\|_\infty.\end{aligned}$$

(c) *Nonnegativity.* $\|x\|_\infty$ is the largest of n nonnegative numbers $|x_k|$.

(d) *Definiteness.* $\|x\|_\infty = 0$ only if $|x_k| = 0$ for $k = 1, \ldots, n$.

3.6 *Taylor approximation of norm.* Find a general formula for the Taylor approximation of the function $f(x) = \|x\|$ near a given nonzero vector z. You can express the approximation in the form $\hat{f}(x) = a^T(x - z) + b$.

Solution.

$$\nabla f(z) = \frac{1}{\sqrt{z_1^2 + \cdots + z_n^2}} \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} = \frac{1}{\|z\|} z.$$

Therefore

$$\begin{aligned} \hat{f}(x) &= \|z\| + \frac{1}{\|z\|} z^T(x - z) \\ &= \|z\| + \frac{1}{\|z\|}(z^T x - z^T z)) \\ &= \|z\| - \|z\| + \frac{1}{\|z\|} z^T x \\ &= \frac{1}{\|z\|} z^T x. \end{aligned}$$

3.7 *Chebyshev inequality.* Suppose x is a 100-vector with $\mathbf{rms}(x) = 1$. What is the maximum number of entries of x that can satisfy $|x_i| \geq 3$? If your answer is k, explain why no such vector can have $k + 1$ entries with absolute values at least 3, and give an example of a specific 100-vector that has RMS value 1, with k of its entries larger than 3 in absolute value.

Solution. By Chebyshev's inequality, the number of entries that satisfy $|x_i| \geq 3$ is no more than $\|x\|^2/3^2 = 100/9$. It follows that we can't have 12 or more entries with absolute value ≥ 3. Can we have 11 entries ≥ 3? The answer is yes: take $x = (3\mathbf{1}_{11}, 1, 0_{88})$.

3.8 *Converse Chebyshev inequality.* Show that at least one entry of a vector has absolute value at least as large as the RMS value of the vector.

Solution. Suppose x is an n-vector, and all its entries have absolute value less than a. Then $\|x\|^2 = \sum_i x_i^2 < \sum_i a^2 = na^2$, so $\|x\| < a\sqrt{n}$. Therefore $\mathbf{rms}(x) = \|x\|/\sqrt{n} < a$. It follows that at least one entry of x has absolute value at least equal to its RMS value.

3.9 *Difference of squared distances.* Determine whether the difference of the squared distances to two fixed vectors c and d, defined as

$$f(x) = \|x - c\|^2 - \|x - d\|^2,$$

is linear, affine, or neither. If it is linear, give its inner product representation, *i.e.*, an n-vector a for which $f(x) = a^T x$ for all x. If it is affine, give a and b for which $f(x) = a^T x + b$ holds for all x. If it is neither linear nor affine, give specific x, y, α, and β for which superposition fails, *i.e.*,

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

(Provided $\alpha + \beta = 1$, this shows the function is neither linear nor affine.)

Solution. *Affine.* (Which is surprising because it involves norms and squares.) Working out the squares gives

$$\begin{aligned} f(x) &= (x - c)^T(x - c) - (x - d)^T(x - d) \\ &= (x^T x - c^T x - x^T c + c^T c) - (x^T x - d^T x - x^T d + d^T d) \\ &= x^T x - 2c^T x + c^T c - x^T x + 2d^T x - d^T d \\ &= 2(d - c)^T x + \|c\|^2 - \|d\|^2. \end{aligned}$$

Hence, f can be expressed as $f(x) = a^T x + b$ with $a = 2(d - c)$ and $b = \|c\|^2 - \|d\|^2$. The function is linear if $\|c\| = \|d\|$.

3.10 *Nearest neighbor document.* Consider the 5 Wikipedia pages in table 3.1 on page 51. What is the nearest neighbor of (the word count histogram vector of) 'Veterans Day' among the others? Does the answer make sense?

Solution. Its nearest neighbor is the histogram vector associated with 'Memorial Day'. (We can see this by scanning the first row or column of table 3.1.) Yes, it makes sense — these two pages describe holidays, whereas the others do not.

3.11 *Neighboring electronic health records.* Let $x_1, \ldots, x_N$ be n-vectors that contain n features extracted from a set of N electronic health records (EHRs), for a population of N patients. (The features might involve patient attributes and current and past symptoms, diagnoses, test results, hospitalizations, procedures, and medications.) Briefly describe in words a practical use for identifying the 10 nearest neighbors of a given EHR (as measured by their associated feature vectors), among the other EHRs.

Solution. If the features are chosen well, with good choice of units for their entries (see page 51), then we can expect the 10 nearest neighbors to involve patients with similar medical issues and conditions. A doctor might look over these other EHRs to see what was done for these similar patients, what the outcomes were, and so on.

3.12 *Nearest point to a line.* Let a and b be different n-vectors. The line passing through a and b is given by the set of vectors of the form $(1 - \theta)a + \theta b$, where θ is a scalar that determines the particular point on the line. (See page 18.)

Let x be any n-vector. Find a formula for the point p on the line that is closest to x. The point p is called the *projection* of x onto the line. Show that $(p - x) \perp (a - b)$, and draw a simple picture illustrating this in 2-D. *Hint.* Work with the square of the distance between a point on the line and x, *i.e.*, $\|(1 - \theta)a + \theta b - x\|^2$. Expand this, and minimize over θ.

Solution. The squared distance of p to x is

$$\begin{aligned} \|(1-\theta)a + \theta b - x\|^2 &= \|a - x + \theta(b-a)\|^2 \\ &= \|a-x\|^2 + 2\theta(a-x)^T(b-a) + \theta^2\|b-a\|^2. \end{aligned}$$

This is a quadratic function of θ. We can minimize it using ordinary calculus by setting the derivative to zero:

$$2(a-x)^T(b-a) + 2\theta\|b-a\|^2 = 0,$$

which yields $\theta = (x-a)^T(b-a)/\|b-a\|^2$. (Recall that $b \neq a$, so the denominator is nonzero.) This yields the closest point on the line,

$$p = a + \frac{(x-a)^T(b-a)}{\|b-a\|^2}(b-a).$$

Let us show that $(p - x) \perp (a - b)$.

$$\begin{aligned} (p-x)^T(a-b) &= \left(a + \frac{(x-a)^T(b-a)}{\|b-a\|^2}(b-a) - x\right)^T (a-b) \\ &= (a-x)^T(a-b) + \frac{(x-a)^T(b-a)}{\|b-a\|^2}(b-a)^T(a-b) \\ &= (a-x)^T(a-b) - \frac{(x-a)^T(b-a)}{\|b-a\|^2}\|b-a\|^2 \\ &= (a-x)^T(a-b) - (x-a)^T(b-a) \\ &= 0. \end{aligned}$$

On the third line we used $(b-a)^T(a-b) = -\|b-a\|^2$

3.13 *Nearest nonnegative vector.* Let x be an n-vector and define y as the nonnegative vector (*i.e.*, the vector with nonnegative entries) closest to x. Give an expression for the elements of y. Show that the vector $z = y - x$ is also nonnegative and that $z^T y = 0$.

Solution.

(a) We find y by minimizing

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

Each term in this sum can be minimized independently of the other terms. The nonnegative y_k that minimizes $(y_k - x_k)^2$ is equal to $y_k = x_k$ if $x_k \geq 0$ and equal to zero if $x_k \leq 0$. Therefore $y_k = \max\{x_k, 0\}$.

(b) We have

$$z_k = y_k - x_k = \begin{cases} 0 & x_k \geq 0 \\ -x_k & x_k < 0. \end{cases}$$

In other words, $z_k = \max\{0, -x_k\}$.

(c) Each term $y_k z_k$ in the inner product of y and z is zero: $y_k = 0$ if $x_k \leq 0$ and $z_k = 0$ if $x_k \geq 0$.

3.14 *Nearest unit vector.* What is the nearest neighbor of the n-vector x among the unit vectors $e_1, \ldots, e_n$?

Solution. We are asked to find the smallest among $\|x - e_i\|^2$, $i = 1, \ldots, n$. We can write this as

$$\|x - e_i\|^2 = \|x\|^2 - 2x^T e_i + 1 = \|x\|^2 + 1 - 2x_i.$$

As the index i varies, the first two terms are constant. To minimize the third term, we choose i for which $x_i = \max_j x_j$. (If there are ties for the max, we can choose any of indices for which $x_i = \max_j x_j$.)

3.15 *Average, RMS value, and standard deviation.* Use the formula (3.5) to show that for any vector x, the following two inequalities hold:

$$|\,\mathbf{avg}(x)| \leq \mathbf{rms}(x), \qquad \mathbf{std}(x) \leq \mathbf{rms}(x).$$

Is it possible to have equality in these inequalities? If $|\,\mathbf{avg}(x)| = \mathbf{rms}(x)$ is possible, give the conditions on x under which it holds. Repeat for $\mathbf{std}(x) = \mathbf{rms}(x)$.

Solution. The inequalities follow immediately from

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

We have an equality $|\,\mathbf{avg}(x)| = \mathbf{rms}(x)$ if $\mathbf{std}(x) = 0$, *i.e.*, x is a constant vector. We have an equality $\mathbf{std}(x) = \mathbf{rms}(x)$ if $\mathbf{avg}(x) = 0$.

3.16 *Effect of scaling and offset on average and standard deviation.* Suppose x is an n-vector and α and β are scalars.

(a) Show that $\mathbf{avg}(\alpha x + \beta \mathbf{1}) = \alpha\,\mathbf{avg}(x) + \beta$.

(b) Show that $\mathbf{std}(\alpha x + \beta \mathbf{1}) = |\alpha|\,\mathbf{std}(x)$.

Solution. Suppose x is an n-vector and α, β are scalars.

(a) First we find the average of $\alpha x + \beta \mathbf{1}$:

$$\mathbf{avg}(\alpha x + \beta \mathbf{1}) = \mathbf{1}^T(\alpha x + \beta \mathbf{1})/n = (\alpha \mathbf{1}^T x + \beta \mathbf{1}^T \mathbf{1})/n = \alpha\,\mathbf{avg}(x) + \beta,$$

where we use $\mathbf{1}^T \mathbf{1} = n$.

(b) Using the definition of $\mathbf{std}(x)$ and part (a),

$$\begin{aligned}\mathbf{std}(\alpha x+\beta\mathbf{1}) &= \mathbf{rms}(\alpha x+\beta\mathbf{1}-(\alpha\,\mathbf{avg}(x)+\beta)\mathbf{1})\\ &= \mathbf{rms}(\alpha x-\alpha\,\mathbf{avg}(x)\mathbf{1})\\ &= |\alpha|\,\mathbf{rms}(x-\mathbf{avg}(x)\mathbf{1})\\ &= |\alpha|\,\mathbf{std}(x).\end{aligned}$$

3.17 *Average and standard deviation of linear combination.* Let $x_1,\ldots,x_k$ be n-vectors, and $\alpha_1,\ldots,\alpha_k$ be numbers, and consider the linear combination $z=\alpha_1x_1+\cdots+\alpha_kx_k$.

(a) Show that $\mathbf{avg}(z)=\alpha_1\,\mathbf{avg}(x_1)+\cdots+\alpha_k\,\mathbf{avg}(x_k)$.

(b) Now suppose the vectors are uncorrelated, which means that for $i\neq j$, x_i and x_j are uncorrelated. Show that $\mathbf{std}(z)=\sqrt{\alpha_1^2\,\mathbf{std}(x_1)^2+\cdots+\alpha_k^2\,\mathbf{std}(x_k)^2}$.

Solution.

(a) The ith entry of z is $z_i=\alpha_1x_{1i}+\alpha_2x_{2i}+\cdots+\alpha_kx_{ki}$, where x_{ji} denotes entry i of vector x_j. Therefore

$$\begin{aligned}\mathbf{avg}(z) &= \frac{z_1+\cdots+z_n}{n}\\ &= \frac{(\alpha_1x_{11}+\cdots+\alpha_kx_{k1})+\cdots+(\alpha_1x_{1n}+\cdots+\alpha_kx_{kn})}{n}\\ &= \alpha_1\frac{x_{11}+\cdots+x_{1n}}{n}+\alpha_2\frac{x_{21}+\cdots+x_{1n}}{n}+\cdots+\alpha_k\frac{x_{k1}+\cdots+\alpha_kx_{kn}}{n}\\ &= \alpha_1\,\mathbf{avg}(x_1)+\cdots+\alpha_k\,\mathbf{avg}(x_k).\end{aligned}$$

(b) We use the notation $\tilde{x}_i=x_i-\mathbf{avg}(x_i)\mathbf{1}$ for entries of the de-meaned vector x_i.

$$\begin{aligned}\|z-\mathbf{avg}(z)\mathbf{1}\|^2 &= \|\alpha_1x_1+\cdots+\alpha_kx_k-(\alpha_1\,\mathbf{avg}(x_1)+\cdots+\alpha_k\,\mathbf{avg}(x_k))\mathbf{1}\|^2\\ &= \|\alpha_1\tilde{x}_1+\cdots+\alpha_k\tilde{x}_k\|^2\\ &= (\alpha_1\tilde{x}_1+\cdots+\alpha_k\tilde{x}_k)^T(\alpha_1\tilde{x}_1+\cdots+\alpha_k\tilde{x}_k)\\ &= \alpha_1^2\|\tilde{x}_1\|^2+\cdots+\alpha_k^2\|\tilde{x}_k\|^2.\end{aligned}$$

The last line follows because all cross terms $\tilde{x}_i^T\tilde{x}_j$ with $i\neq j$ are zero, since the vectors are uncorrelated. Therefore

$$\begin{aligned}\mathbf{std}(z)^2 &= \frac{1}{n}\|z-\mathbf{avg}(z)\mathbf{1}\|^2\\ &= \alpha_1^2\frac{\|\tilde{x}_1\|^2}{n}+\alpha_2^2\frac{\|\tilde{x}_2\|^2}{n}+\cdots+\alpha_k^2\frac{\|\tilde{x}_k\|^2}{n}\\ &= \alpha_1^2\,\mathbf{std}(x_1)^2+\cdots+\alpha_k^2\,\mathbf{std}(x_k)^2.\end{aligned}$$

3.18 *Triangle equality.* When does the triangle inequality hold with equality, *i.e.*, what are the conditions on a and b to have $\|a+b\|=\|a\|+\|b\|$?

Solution. We have $\|a+b\|=\|a\|+\|b\|$ if $a=0$ or $b=0$, or if a and b are nonzero and positive multiples of each other.

The triangle inequality can be proved as follows:

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

In step 4 we use the Cauchy–Schwarz inequality. We see that the triangle inequality holds with equality if $a^T b = \|a\|\|b\|$. For nonzero vectors, this means that a and b are positive multiples of each other.

3.19 *Norm of sum.* Use the formulas (3.1) and (3.6) to show the following:

(a) $a \perp b$ if and only if $\|a+b\| = \sqrt{\|a\|^2 + \|b\|^2}$.

(b) Nonzero vectors a and b make an acute angle if and only if $\|a+b\| > \sqrt{\|a\|^2 + \|b\|^2}$.

(c) Nonzero vectors a and b make an obtuse angle if and only if $\|a+b\| < \sqrt{\|a\|^2 + \|b\|^2}$.

Draw a picture illustrating each case in 2-D.

Solution. The three statements follow from the identity

$$\|a+b\|^2 = \|a\|^2 + 2a^T b + \|b\|^2.$$

If a and b are nonzero, this can be written as

$$\|a+b\|^2 = \|a\|^2 + 2\cos\theta\|a\|\|b\| + \|b\|^2.$$

where $\theta = \angle(a,b)$.

3.20 *Regression model sensitivity.* Consider the regression model $\hat{y} = x^T\beta + v$, where $\hat{y}$ is the prediction, x is a feature vector, β is a coefficient vector, and v is the offset term. If x and $\tilde{x}$ are feature vectors with corresponding predictions $\hat{y}$ and $\tilde{y}$, show that $|\hat{y}-\tilde{y}| \leq \|\beta\|\|x-\tilde{x}\|$. This means that when $\|\beta\|$ is small, the prediction is not very sensitive to a change in the feature vector.

Solution. The inequality follows from the Cauchy–Schwarz inequality:

$$|y-\tilde{y}| = |\beta^T x + v - (\beta^T\tilde{x} + v)| = |\beta^T(x-\tilde{x})| \leq \|\beta\|\|x-\tilde{x}\|.$$

3.21 *Dirichlet energy of a signal.* Suppose the T-vector x represents a time series or signal. The quantity

$$\mathcal{D}(x) = (x_1 - x_2)^2 + \cdots + (x_{T-1} - x_T)^2,$$

the sum of the differences of adjacent values of the signal, is called the *Dirichlet energy* of the signal, named after the mathematician Peter Gustav Lejeune Dirichlet. The Dirichlet energy is a measure of the roughness or wiggliness of the time series. It is sometimes divided by $T-1$, to give the mean square difference of adjacent values.

(a) Express $\mathcal{D}(x)$ in vector notation. (You can use vector slicing, vector addition or subtraction, inner product, norm, and angle.)

(b) How small can $\mathcal{D}(x)$ be? What signals x have this minimum value of the Dirichlet energy?

(c) Find a signal x with entries no more than one in absolute value that has the largest possible value of $\mathcal{D}(x)$. Give the value of the Dirichlet energy achieved.

Solution.

(a) $\mathcal{D} = \|x_{1:T-1} - x_{2:T}\|^2$.

(b) The smallest possible value is 0. This is achieved by any constant signal, *i.e.*, one with all entries equal. (These are the least rough, or smoothest, signals.)

(c) The largest possible value of $\mathcal{D}$, for $|x_i| \leq 1$, is attained when $x_i = (-1)^i$ (or $x_i = -(-1)^i$). This makes each term in the sum equal to 4, the largest possible value, and we have $\mathcal{D} = 4(T-1)$. These are the wiggliest or roughest signals that satisfy $|x_i| \leq 1$.

3.22 *Distance from Palo Alto to Beijing.* The surface of the earth is reasonably approximated as a sphere with radius $R = 6367.5$km. A location on the earth's surface is traditionally given by its latitude θ and its longitude λ, which correspond to angular distance from the equator and prime meridian, respectively. The 3-D coordinates of the location are given by

$$\begin{bmatrix} R\sin\lambda\cos\theta \\ R\cos\lambda\cos\theta \\ R\sin\theta \end{bmatrix}.$$

(In this coordinate system $(0,0,0)$ is the center of the earth, $R(0,0,1)$ is the North pole, and $R(0,1,0)$ is the point on the equator on the prime meridian, due south of the Royal Observatory outside London.)

The distance *through the earth* between two locations (3-vectors) a and b is $\|a-b\|$. The distance *along the surface of the earth* between points a and b is $R\angle(a,b)$. Find these two distances between Palo Alto and Beijing, with latitudes and longitudes given below.

City	Latitude θ	Longitude λ
Beijing	39.914°	116.392°
Palo Alto	37.429°	-122.138°

Solution. Let a and b be the 3-D coordinates of Palo Alto and Beijing, respectively. We can compute a and b from the given formula:

$$a = (4374.9, -2171.0, 4085.6), \qquad b = (-4281.7, -2689.8, 3870.0).$$

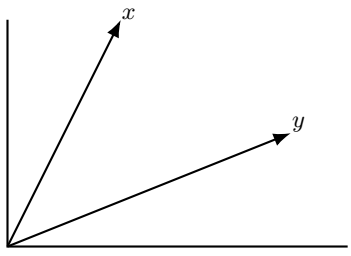
The distance through the earth is $\|a-b\| = 8674.8$ km. The angle between a and b can be derived using the inner product. Recall that $a^Tb = \|a\|\|b\|\cos(\angle(a,b))$. Solving for $\angle(a,b)$ we find that

$$\angle(a,b) = \arccos\left(\frac{a^Tb}{\|a\|\|b\|}\right) = \arccos\left(\frac{a^Tb}{R^2}\right) = 1.4987 \text{ radians}.$$

Multiplying $\angle(a,b)$ by R gives the distance along the surface of the earth, 9543.2 km.

3.23 *Angle between two nonnegative vectors.* Let x and y be two nonzero n-vectors with nonnegative entries, *i.e.*, each $x_i \geq 0$ and each $y_i \geq 0$. Show that the angle between x and y lies between 0 and 90°. Draw a picture for the case when $n = 2$, and give a short geometric explanation. When are x and y orthogonal?

Solution. The angle lies between 0 and 90° because the inner product of two elementwise nonnegative vectors, which is a sum of products of nonnegative numbers, is nonnegative. When $n = 2$, the two vectors lie in the positive orthant, also known as quadrant I. The maximum angle between any such vectors is clearly 90°. This is illustrated below.



Two nonnegative vectors are orthogonal if and only if they have a complementary zero-nonzero pattern, *i.e.*, $x_k = 0$ whenever $y_k > 0$ and $y_k = 0$ whenever $x_k > 0$.

3.24 *Distance versus angle nearest neighbor.* Suppose $z_1, \ldots, z_m$ is a collection of n-vectors, and x is another n-vector. The vector z_j is the (distance) nearest neighbor of x (among the given vectors) if

$$\|x - z_j\| \leq \|x - z_i\|, \quad i = 1, \ldots, m,$$

i.e., x has smallest distance to z_j. We say that z_j is the *angle nearest neighbor* of x if

$$\angle(x, z_j) \leq \angle(x, z_i), \quad i = 1, \ldots, m,$$

i.e., x has smallest angle to z_j.

(a) Give a simple specific numerical example where the (distance) nearest neighbor is not the same as the angle nearest neighbor.

(b) Now suppose that the vectors $z_1, \ldots, z_m$ are normalized, which means that $\|z_i\| = 1$, $i = 1, \ldots, m$. Show that in this case the distance nearest neighbor and the angle nearest neighbor are always the same. *Hint.* You can use the fact that arccos is a decreasing function, *i.e.*, for any u and v with $-1 \leq u < v \leq 1$, we have $\arccos(u) > \arccos(v)$.

Solution.

(a) We can even create an example with $n = 1$! Let's take $z_1 = 1$, $z_2 = -10$, and $x = -1$. Then we have

$$\|x - z_1\| = 2, \quad \|x - z_2\| = 9, \quad \angle(x, z_1) = \pi/2, \quad \angle(x, z_2) = 0,$$

so the distance nearest neighbor is z_1 but the angle nearest neighbor is z_2. You can of course create examples in higher dimensions too.

(b) Now we assume that $\|z_i\| = 1$ for $i = 1, \ldots, m$. We have

$$\begin{aligned}
\|x - z_j\| \leq \|x - z_i\| &\iff \|x - z_j\|^2 \leq \|x - z_i\|^2 \\
&\iff \|x\|^2 - 2x^T z_j + \|z_j\|^2 \leq \|x\|^2 - 2x^T z_i + \|z_i\|^2 \\
&\iff -2x^T z_j \leq -2x^T z_i \\
&\iff x^T z_j \geq x^T z_i \\
&\iff \frac{x^T z_j}{\|x\|\|z_j\|} \geq \frac{x^T z_i}{\|x\|\|z_i\|} \\
&\iff \arccos\left(\frac{x^T z_j}{\|x\|\|z_j\|}\right) \leq \arccos\left(\frac{x^T z_i}{\|x\|\|z_i\|}\right) \\
&\iff \angle(x, z_j) \leq \angle(x, z_i).
\end{aligned}$$

Let's justify some of these lines. In the third line, we cancel $\|x\|^2$ from both sides, and also $\|z_j\|^2$ and $\|z_i\|^2$, since they are both one. In the 5th line, we use the fact that $\|z_j\| = \|z_i\| = 1$. In the 6th line, we use the fact that the arc-cosine function is decreasing, *i.e.*, if $a \geq b$, then $\arccos(a) \leq \arccos(b)$.

3.25 *Leveraging.* Consider an asset with return time series over T periods given by the T-vector r. This asset has mean return μ and risk σ, which we assume is positive. We also consider cash as an asset, with return vector $\mu^{\mathrm{rf}}\mathbf{1}$, where μ^{rf} is the cash interest rate per period. Thus, we model cash as an asset with return μ^{rf} and zero risk. (The superscript in μ^{rf} stands for 'risk-free'.) We will create a simple portfolio consisting of the asset and cash. If we invest a fraction θ in the asset, and $1 - \theta$ in cash, our portfolio return is given by the time series

$$p = \theta r + (1 - \theta)\mu^{\mathrm{rf}}\mathbf{1}.$$

We interpret θ as the fraction of our portfolio we hold in the asset. We allow the choices $\theta > 1$, or $\theta < 0$. In the first case we are *borrowing* cash and using the proceeds to buy more of the asset, which is called *leveraging*. In the second case we are *shorting* the asset. When θ is between 0 and 1 we are blending our investment in the asset and cash, which is a form of *hedging*.

(a) Derive a formula for the return and risk of the portfolio, *i.e.*, the mean and standard deviation of p. These should be expressed in terms of μ, σ, μ^{rf}, and θ. Check your formulas for the special cases $\theta = 0$ and $\theta = 1$.

(b) Explain how to choose θ so the portfolio has a given target risk level σ^{tar} (which is positive). If there are multiple values of θ that give the target risk, choose the one that results in the highest portfolio return.

(c) Assume we choose the value of θ as in part (b). When do we use leverage? When do we short the asset? When do we hedge? Your answers should be in English.

Solution.

(a) The mean return of the portfolio is the average of the vector p:

$$\begin{aligned} \mathbf{avg}\,(p) &= \mathbf{avg}\,(\theta r + (1-\theta)\mu^{\mathrm{rf}}\mathbf{1}) \\ &= \theta\,\mathbf{avg}(r) + (1-\theta)\mu^{\mathrm{rf}}\,\mathbf{avg}(\mathbf{1}) \\ &= \theta\mu + (1-\theta)\mu^{\mathrm{rf}}. \end{aligned}$$

On the last line we use $\mathbf{avg}(r) = \mu$, and $\mathbf{avg}(\mathbf{1}) = 1$.

The risk is the standard deviation of the vector p:

$$\begin{aligned} \mathbf{std}(p) &= \|p - \mathbf{avg}(p)\mathbf{1}\|/\sqrt{T} \\ &= \|\theta r + (1-\theta)\mu^{\mathrm{rf}}\mathbf{1} - (\theta\mu + (1-\theta)\mu^{\mathrm{rf}})\mathbf{1}\|/\sqrt{T} \\ &= \|\theta(r - \mu\mathbf{1})\|/\sqrt{T} \\ &= |\theta|\|r - \mu\mathbf{1}\|/\sqrt{T} \\ &= |\theta|\,\mathbf{std}(r) \\ &= |\theta|\sigma. \end{aligned}$$

On line 2 we use the expression for $\mathbf{avg}(p)$ that we derived in part (a). The last step is the definition of $\sigma = \mathbf{std}(r)$.

(b) To achieve the target risk σ^{tar}, we need $|\theta| = \sigma^{\mathrm{tar}}/\sigma$, so there are two choices:

$$\theta = \sigma^{\mathrm{tar}}/\sigma, \qquad \theta = -\sigma^{\mathrm{tar}}/\sigma.$$

To choose the sign of θ we consider the portfolio return

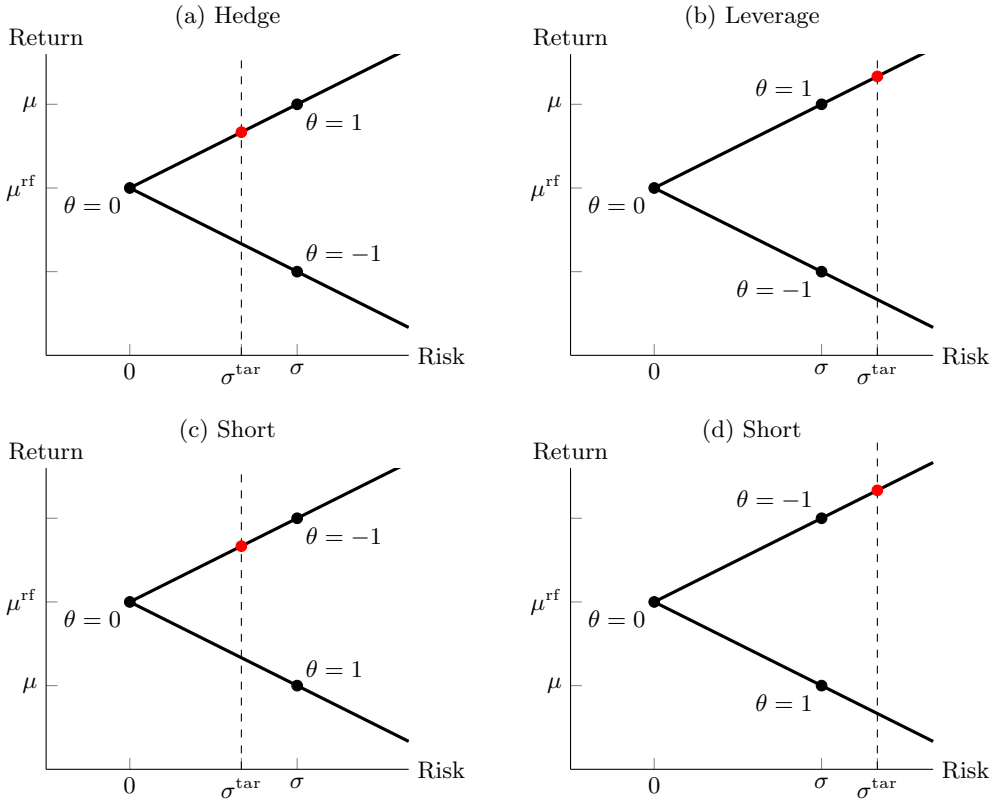
$$\theta\mu + (1-\theta)\mu^{\mathrm{rf}} = \mu^{\mathrm{rf}} + \theta(\mu - \mu^{\mathrm{rf}}).$$

To maximize this, for given $|\theta|$, we choose θ positive if $\mu > \mu^{\mathrm{rf}}$ and θ negative if $\mu < \mu^{\mathrm{rf}}$. This means we short the asset when its return is less than the risk-free return.

(c) We can distinguish the cases shown in the figure below. The solid lines show

$$(\text{risk, return}) = (\mathbf{std}(p), \mathbf{avg}(p)) = (|\theta|\sigma, \mu^{\mathrm{rf}} + \theta(\mu - \mu^{\mathrm{rf}}))$$

for all values of θ. The red dot shows the portfolio with the hightest return for the given target valu of risk.



In case (a), the asset return is more than the risk-free return ($\mu > \mu^{\mathrm{rf}}$) and its risk is higher than the target risk ($\sigma > \sigma^{\mathrm{tar}}$). In this case we hedge ($0 < \theta < 1$). In case (b), the asset return is more than the risk-free return ($\mu > \mu^{\mathrm{rf}}$) and its risk is less than the target risk ($\sigma < \sigma^{\mathrm{tar}}/\sigma$). In this case we leverage ($\theta > 1$). In cases (c) and (d), the asset return is less than the risk-free return ($\mu < \mu^{\mathrm{f}}$). In this case we short ($\theta < 0$).

To summarize, we short the asset when its return is less than the risk-free return. We hedge when the asset return is more than the risk-free return and the asset risk is higher than the target risk. We leverage when the asset return is more than the risk-free return and the asset risk is less than the target risk.

3.26 *Time series auto-correlation.* Suppose the T-vector x is a non-constant time series, with x_t the value at time (or period) t. Let $\mu = (\mathbf{1}^T x)/T$ denote its mean value. The *auto-correlation* of x is the function $R(\tau)$, defined for $\tau = 0, 1, \ldots$ as the correlation coefficient of the two vectors $(x, \mu\mathbf{1}_\tau)$ and $(\mu\mathbf{1}_\tau, x)$. (The subscript τ denotes the length of the ones vector.) Both of these vectors also have mean μ. Roughly speaking, $R(\tau)$ tells us how correlated the time series is with a version of itself lagged or shifted by τ periods. (The argument τ is called the lag.)

(a) Explain why $R(0) = 1$, and $R(\tau) = 0$ for $\tau \geq T$.

(b) Let z denote the standardized or z-scored version of x (see page 56). Show that for $\tau = 0, \ldots, T-1$,

$$R(\tau) = \frac{1}{T} \sum_{t=1}^{T-\tau} z_t z_{t+\tau}.$$

(c) Find the auto-correlation for the time series $x = (+1, -1, +1, -1, \ldots, +1, -1)$. You can assume that T is even.

(d) Suppose x denotes the number of meals served by a restaurant on day τ. It is observed that $R(7)$ is fairly high, and $R(14)$ is also high, but not as high. Give an English explanation of why this might be.

Solution.

(a) $R(0)$ is the correlation coefficient between x and x, which is always one.
To find $R(\tau)$, we consider the vectors $(\mu\mathbf{1}_\tau, x)$ and $(x, \mu\mathbf{1}_\tau)$. They each have mean μ, so their de-meaned versions are $(0_\tau, x - \mu\mathbf{1})$ and $(x - \mu\mathbf{1}, 0_\tau)$. The two vectors have the same norm $\|x - \mu\mathbf{1}\|$. Therefore their correlation coefficient is

$$R(\tau) = \frac{(0_\tau, x - \mu\mathbf{1})^T (x - \mu\mathbf{1}, 0_\tau)}{\|x - \mu\mathbf{1}\|^2}.$$

For $\tau \geq T$, the inner product in the numerator is zero, since for each i, one of the two vectors in the inner product has ith entry zero. For $\tau = 0, \ldots, T-1$, the expression for $R(\tau)$ reduces to

$$R(\tau) = \frac{\sum_{t=1}^{T-\tau} (x_t - \mu)(x_{t+\tau} - \mu)}{\sum_{t=1}^{T} (x_t - \mu)^2}.$$

(b) We express the formula above as the inner product

$$\begin{aligned} R(\tau) &= \left(\frac{(0_\tau, x - \mu\mathbf{1})}{\|x - \mu\mathbf{1}\|}\right)^T \left(\frac{(x - \mu\mathbf{1}, 0_\tau)}{\|x - \mu\mathbf{1}\|}\right) \\ &= \left(\frac{(0_\tau, x - \mu\mathbf{1})}{\sqrt{T}\,\mathbf{std}(x)}\right)^T \left(\frac{(x - \mu\mathbf{1}, 0_\tau)}{\sqrt{T}\,\mathbf{std}(x)}\right) \\ &= \frac{1}{T}(0_\tau, z)^T (z, 0_\tau). \end{aligned}$$

In this inner product we do not need to include the sum over the first τ entries (since the first vector has zero entries), and we do not need to sum over the last τ entries (since the second vector has zero entries). Summing over the remaining indices we get

$$R(\tau) = \frac{1}{T} \sum_{t=1}^{T-\tau} z_t z_{t+\tau}.$$

(c) The time series $x = (+1, -1, \ldots, +1, -1)$ has mean zero and norm $\sqrt{T}$. For $\tau = 0, \ldots, T-1$, the auto-correlation is

$$\begin{aligned} R(\tau) &= \frac{1}{T} \sum_{t=1}^{T-\tau} x_t x_{t+\tau} \\ &= \frac{1}{T} \sum_{t=1}^{T-\tau} (-1)^{t+1} (-1)^{t+\tau+1} \\ &= \frac{T-\tau}{T} (-1)^\tau \\ &= \begin{cases} 1 - \tau/T & \tau \text{ even} \\ -1 + \tau/T & \tau \text{ odd.} \end{cases} \end{aligned}$$

(d) $R(7)$ large (*i.e.*, near one) means that x_t and x_{t+7} are often above or below the mean value together. x_{t+7} is the number of meals served exactly one week after x_t, so this means that, for example, saturdays are often above the mean together, and tuesdays are often below the mean together.

3.27 *Another measure of the spread of the entries of a vector.* The standard deviation is a measure of how much the entries of a vector differ from their mean value. Another measure of how much the entries of an n-vector x differ from each other, called the *mean square difference*, is defined as

$$\mathrm{MSD}(x) = \frac{1}{n^2}\sum_{i,j=1}^{n}(x_i - x_j)^2.$$

(The sum means that you should add up the n^2 terms, as the indices i and j each range from 1 to n.) Show that $\mathrm{MSD}(x) = 2\,\mathbf{std}(x)^2$. *Hint.* First observe that $\mathrm{MSD}(\tilde{x}) = \mathrm{MSD}(x)$, where $\tilde{x} = x - \mathbf{avg}(x)\mathbf{1}$ is the de-meaned vector. Expand the sum and recall that $\sum_{i=1}^{n}\tilde{x}_i = 0$.

Solution. Adding a constant to all the entries of x does not change $\mathrm{MSD}(x)$. Therefore

$$\mathrm{MSD}(x) = \mathrm{MSD}(x - \mathbf{avg}(x)\mathbf{1}) = \mathrm{MSD}(\tilde{x}).$$

Now we work out $\mathrm{MSD}(\tilde{x})$:

$$\begin{aligned}\mathrm{MSD}(\tilde{x}) &= \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(\tilde{x}_i - \tilde{x}_j)^2 \\ &= \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(\tilde{x}_i^2 + \tilde{x}_j^2 - 2\tilde{x}_i\tilde{x}_j).\end{aligned}$$

We look at the three terms in the sum separately. The first two terms are

$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\tilde{x}_i^2 = \frac{1}{n}\sum_{i=1}^{n}\tilde{x}_i^2 = \mathbf{std}(x)^2$$

and

$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\tilde{x}_j^2 = \frac{1}{n}\sum_{j=1}^{n}\tilde{x}_j^2 = \mathbf{std}(x)^2.$$

The third term is zero:

$$\sum_{i=1}^{n}\sum_{j=1}^{n}2\tilde{x}_i\tilde{x}_j = 2\left(\sum_{i=1}^{n}\tilde{x}_i\right)\left(\sum_{j=1}^{n}\tilde{x}_j\right) = 0 \times 0 = 0.$$

So putting it all together we have $\mathrm{MSD}(x) = 2\,\mathbf{std}(x)^2$.

3.28 *Weighted norm.* On page 51 we discuss the importance of choosing the units or scaling for the individual entries of vectors, when they represent heterogeneous quantities. Another approach is to use a *weighted norm* of a vector x, defined as

$$\|x\|_w = \sqrt{w_1x_1^2 + \cdots + w_nx_n^2},$$

where $w_1, \ldots, w_n$ are given positive *weights*, used to assign more or less importance to the different elements of the n-vector x. If all the weights are one, the weighted norm reduces to the usual ('unweighted') norm. It can be shown that the weighted norm is a general norm, *i.e.*, it satisfies the four norm properties listed on page 46. Following the discussion

on page 51, one common rule of thumb is to choose the weight w_i as the inverse of the typical value of x_i^2 in the application.

A version of the Cauchy–Schwarz inequality holds for weighted norms: For any n-vector x and y, we have

$$|w_1x_1y_1 + \cdots + w_nx_ny_n| \leq \|x\|_w\|y\|_w.$$

(The expression inside the absolute value on the left-hand side is sometimes called the weighted inner product of x and y.) Show that this inequality holds. *Hint.* Consider the vectors $\tilde{x} = (x_1\sqrt{w_1}, \ldots, x_n\sqrt{w_n})$ and $\tilde{y} = (y_1\sqrt{w_1}, \ldots, y_n\sqrt{w_n})$, and use the (standard) Cauchy–Schwarz inequality.

Solution. We follow the hint and apply the Cauchy–Schwarz inequality to the vectors $\tilde{x}$ and $\tilde{y}$:

$$\begin{aligned} |w_1x_1y_1 + \cdots + w_nx_ny_n| &= |\tilde{x}^T\tilde{y}| \\ &\leq \|\tilde{x}\|\|\tilde{y}\| \\ &= \sqrt{w_1x_1^2 + \cdots + w_nx_n^2}\sqrt{w_1y_1^2 + \cdots + w_ny_n^2} \\ &= \|\tilde{x}\|_w\|\tilde{y}\|_w. \end{aligned}$$

The second line is the standard Cauchy–Schwarz inequality.

Chapter 4

Clustering

Exercises

4.1 *Minimizing mean square distance to a set of vectors.* Let $x_1, \ldots, x_L$ be a collection of n-vectors. In this exercise you will fill in the missing parts of the argument to show that the vector z which minimizes the sum-square distance to the vectors,

$$J(z) = \|x_1 - z\|^2 + \cdots + \|x_L - z\|^2,$$

is the average or centroid of the vectors, $\overline{x} = (1/L)\,(x_1 + \cdots + x_L)$. (This result is used in one of the steps in the k-means algorithm. But here we have simplified the notation.)

(a) Explain why, for any z, we have

$$J(z) = \sum_{i=1}^{L} \|x_i - \overline{x} - (z - \overline{x})\|^2 = \sum_{i=1}^{L} \left(\|x_i - \overline{x}\|^2 - 2(x_i - \overline{x})^T(z - \overline{x})\right) + L\|z - \overline{x}\|^2.$$

(b) Explain why $\sum_{i=1}^{L}(x_i - \overline{x})^T(z - \overline{x}) = 0$. *Hint.* Write the left-hand side as

$$\left(\sum_{i=1}^{L}(x_i - \overline{x})\right)^T (z - \overline{x}),$$

and argue that the left-hand vector is 0.

(c) Combine the results of (a) and (b) to get $J(z) = \sum_{i=1}^{L} \|x_i - \overline{x}\|^2 + L\|z - \overline{x}\|^2$. Explain why for any $z \neq \overline{x}$, we have $J(z) > J(\overline{x})$. This shows that the choice $z = \overline{x}$ minimizes $J(z)$.

Solution. There's not much to do in this problem.

(a) In the second expression, are simply adding and subtracting the vector $\overline{x}$ from $x_i - z$, which has no effect. Expanding the norm squared gives the right-hand expression.

(b) We follow the hint, which works since the inner product is a linear function, so

$$\sum_{i=1}^{L}(x_i - \overline{x})^T(z - \overline{x}) = \left(\sum_{i=1}^{L}(x_i - \overline{x})\right)^T (z - \overline{x}).$$

(Note that the first sum is over numbers, and the second is over vectors.) Note also that the right-hand vector in the inner product is the same for each i. The vector $\sum_{i=1}^{L}(x_i - \overline{x})$ is zero, since

$$\sum_{i=1}^{L} x_i = L\overline{x} = \sum_{i=1}^{L} \overline{x}.$$

Therefore the middle term in the right-hand expression for $J(z)$ in part (a) is zero.

(c) For any $z \neq \overline{x}$, we have

$$J(z) = \sum_{i=1}^{L} \|x_i - \overline{x}\|^2 + L\|z - \overline{x}\|^2 > \sum_{i=1}^{L} \|x_i - \overline{x}\|^2 = J(\overline{x}),$$

where we use $\|z - \overline{x}\| > 0$.

The result is illustrated in figure 4.1.

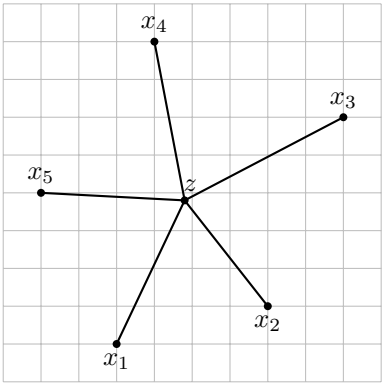


Figure 4.1 The vector z is the mean or centroid of the vectors $x_1, \ldots, x_5$. It minimizes the mean square distance to the points.

4.2 *k-means with nonnegative, proportions, or Boolean vectors.* Suppose that the vectors $x_1, \ldots, x_N$ are clustered using k-means, with group representatives $z_1, \ldots, z_k$.

(a) Suppose the original vectors x_i are nonnegative, *i.e.*, their entries are nonnegative. Explain why the representatives z_j are also nonnegative.

(b) Suppose the original vectors x_i represent proportions, *i.e.*, their entries are nonnegative and sum to one. (This is the case when x_i are word count histograms, for example.) Explain why the representatives z_j also represent proportions, *i.e.*, their entries are nonnegative and sum to one.

(c) Suppose the original vectors x_i are Boolean, *i.e.*, their entries are either 0 or 1. Give an interpretation of $(z_j)_i$, the ith entry of the j group representative.

Hint. Each representative is the average of some of the original vectors.

Solution. The group representative z_j is the average of the vectors x_k in the group:

$$z_j = \frac{1}{|G_j|} \sum_{k \in G_j} x_k.$$

(a) If the vectors x_k are nonnegative, the average z_j is a nonnegative vector.

(b) If each vector sums to one, $\mathbf{1}^T x_k = 1$ for all k, then the same is true for the average:

$$\mathbf{1}^T z_j = \frac{1}{|G_j|} \sum_{k \in G_j} \mathbf{1}^T x_k = \frac{|G_j|}{|G_j|} = 1.$$

(c) The ith entry of group representative z_j is the fraction of the vectors in group j that have ith entry one. If it is equal to one, all vectors in the group have ith entry one. If it is close to one, most vectors in the group have ith entry one. If it zero, no vectors in the group have ith entry one.

4.3 *Linear separation in 2-way partitioning.* Clustering a collection of vectors into $k = 2$ groups is called 2-way partitioning, since we are partitioning the vectors into 2 groups, with index sets G_1 and G_2. Suppose we run k-means, with $k = 2$, on the n-vectors $x_1, \ldots, x_N$. Show that there is a nonzero vector w and a scalar v that satisfy

$$w^T x_i + v \geq 0 \text{ for } i \in G_1, \qquad w^T x_i + v \leq 0 \text{ for } i \in G_2.$$

In other words, the affine function $f(x) = w^T x + v$ is greater than or equal to zero on the first group, and less than or equal to zero on the second group. This is called *linear separation* of the two groups (although *affine separation* would be more accurate).

Hint. Consider the function $\|x - z_1\|^2 - \|x - z_2\|^2$, where z_1 and z_2 are the group representatives.

Solution. If $i \in G_1$, then x_i is closer to z_1 than to z_2:

$$\|x_i - z_2\|^2 - \|x_i - z_1\|^2 \geq 0.$$

If $i \in G_2$, then x_i is closer to z_2 than to z_1:

$$\|x_i - z_2\|^2 - \|x_i - z_2\|^2 \leq 0.$$

Expanding the norms on the left-hand side gives

$$\begin{aligned} \|x_i - z_2\|^2 - \|x_i - z_1\|^2 &= \|x_i\|^2 - 2z_2^T x_i + \|z_2\|^2 - \|x_i\|^2 + 2z_1^T x_i - \|z_1\|^2 \\ &= -2(z_2 - z_1)^T x_i + \|z_2\|^2 - \|z_1\|^2. \end{aligned}$$

We see that if we take

$$w = -2(z_2 - z_1), \qquad v = \|z_2\|^2 - \|z_1\|^2,$$

then $w^T x_i + v \geq 0$ for $i \in G_1$ and $w^T x_i + v \leq 0$ for $i \in G_2$.

4.4 *Pre-assigned vectors.* Suppose that some of the vectors $x_1, \ldots, x_N$ are assigned to specific groups. For example, we might insist that x_{27} be assigned to group 5. Suggest a simple modification of the k-means algorithm that respects this requirement. Describe a practical example where this might arise, when each vector represents n features of a medical patient.

Solution. We modify k-means by over-ruling the partitioning in step 1. For any vector that is pre-assigned to a specific group, we use the given assignment.

This might arise when the vectors represent patient feature vectors, and the groups (at least the ones we pre-assign) represent diagnoses of specific diseases. We have a collection of N patient vectors; for some of them, we have a known definitive diagnosis of the disease they have.

Chapter 5

Linear independence

Exercises

5.1 *Linear independence of stacked vectors.* Consider the stacked vectors

$$c_1 = \begin{bmatrix} a_1 \\ b_1 \end{bmatrix}, \quad \ldots, \quad c_k = \begin{bmatrix} a_k \\ b_k \end{bmatrix},$$

where $a_1, \ldots, a_k$ are n-vectors and $b_1, \ldots, b_k$ are m-vectors.

(a) Suppose $a_1, \ldots, a_k$ are linearly independent. (We make no assumptions about the vectors $b_1, \ldots, b_k$.) Can we conclude that the stacked vectors $c_1, \ldots, c_k$ are linearly independent?

(b) Now suppose that $a_1, \ldots, a_k$ are linearly dependent. (Again, with no assumptions about $b_1, \ldots, b_k$.) Can we conclude that the stacked vectors $c_1, \ldots, c_k$ are linearly dependent?

Solution.

(a) Yes, the stacked vectors are always independent. Suppose that $\beta_1 c_1 + \cdots + \beta_k c_k = 0$. Then we have $\beta_1 a_1 + \cdots + \beta_k a_k = 0$ and $\beta_1 b_1 + \cdots + \beta_k b_k = 0$. Since $a_1, \ldots, a_k$ are linearly independent, we must have $\beta_1 = \cdots = \beta_k = 0$. So $c_1, \ldots, c_k$ are linearly independent.

(b) No, we can't conclude that they are dependent. We can find examples with a_i linearly dependent, and c_i linearly independent, and also examples with a_i linearly dependent, and c_i linearly dependent. For the first example, take $a_1 = a_2 = 1$ (they are 1-vectors), and $b_1 = 1$, $b_2 = 0$. For the second example, take the same a_i and $b_1 = b_2 = 1$.

5.2 *A surprising discovery.* An intern at a quantitative hedge fund examines the daily returns of around 400 stocks over one year (which has 250 trading days). She tells her supervisor that she has discovered that the returns of one of the stocks, Google (GOOG), can be expressed as a linear combination of the others, which include many stocks that are unrelated to Google (say, in a different type of business or sector).

Her supervisor then says: "It is overwhelmingly unlikely that a linear combination of the returns of unrelated companies can reproduce the daily return of GOOG. So you've made a mistake in your calculations."

Is the supervisor right? Did the intern make a mistake? Give a very brief explanation.

Solution. The supervisor is wrong, and the intern is most likely correct. She is examining 400 250-vectors, each representing the daily returns of a particular stock over one year. By the independence-dimension inequality, any set of $n + 1$ or more n vectors is linearly dependent, so we know that this set of vectors is linearly dependent. That is, there exist some vectors that can be expressed as a linear combination of the others. It is quite likely that the returns of any particular stock, such as GOOG, can be expressed as a linear combination of the returns other stocks.

Even if Google's return can be expressed as a linear combination of the others, by the independence-dimension inequality, this fact is not as useful as it might seems. For example, Google's future returns would not be given by the same linear combination of other asset returns. So although the intern is right, and the supervisor is wrong, the observation cannot be monetized.

5.3 *Replicating a cash flow with single-period loans.* We continue the example described on page 93. Let c be any n-vector representing a cash flow over n periods. Find the coefficients $\alpha_1, \ldots, \alpha_n$ of c in its expansion in the basis $e_1, l_1, \ldots, l_{n-1}$, *i.e.*,

$$c = \alpha_1 e_1 + \alpha_2 l_1 + \cdots + \alpha_n l_{n-1}.$$

Verify that α_1 is the net present value (NPV) of the cash flow c, defined on page 22, with interest rate r. *Hint.* Use the same type of argument that was used to show that

$e_1, l_1, \ldots, l_{n-1}$ are linearly independent. Your method will find α_n first, then α_{n-1}, and so on.

Solution. As noted on page 93,

$$\alpha_1 e_1 + \alpha_2 l_1 + \cdots + \alpha_n l_{n-1} = \begin{bmatrix} \alpha_1 + \alpha_2 \\ \alpha_3 - (1+r)\alpha_2 \\ \vdots \\ \alpha_n - (1+r)\alpha_{n-1} \\ -(1+r)\alpha_n \end{bmatrix}.$$

We equate this to $c = (c_1, c_2, \ldots, c_{n-1}, c_n)$, and determine α_n, α_{n-1}, $\ldots$, α_1 from the equality:

$$\alpha_n = -\frac{c_n}{1+r},$$

and therefore

$$\alpha_{n-1} = -\frac{c_{n-1} - \alpha_n}{1+r} = -\frac{c_{n-1}}{1+r} - \frac{c_n}{(1+r)^2},$$

and

$$\alpha_{n-2} = -\frac{c_{n-2} - \alpha_{n-1}}{1+r} = -\frac{c_{n-2}}{1+r} - \frac{c_{n-1}}{(1+r)^2} - \frac{c_n}{(1+r)^3},$$

et cetera. Continuing the pattern we find

$$\alpha_2 = -\frac{c_2}{1+r} - \frac{c_3}{(1+r)^2} - \cdots - \frac{c_n}{(1+r)^{n-1}}$$

and finally

$$\alpha_1 = c_1 - \alpha_2 = c_1 + \frac{c_2}{1+r} + \cdots + \frac{c_n}{(1+r)^{n-1}}.$$

This is exactly the net present value (NPV) of the cash flow, with interest rate r.

5.4 *Norm of linear combination of orthonormal vectors.* Suppose $a_1, \ldots, a_k$ are orthonormal n-vectors, and $x = \beta_1 a_1 + \cdots + \beta_k a_k$, where $\beta_1, \ldots, \beta_k$ are scalars. Express $\|x\|$ in terms of $\beta = (\beta_1, \ldots, \beta_k)$.

Solution.

$$\begin{aligned} \|x\|^2 &= x^T x \\ &= (\beta_1 a_1 + \cdots + \beta_k a_k)^T (\beta_1 a_1 + \cdots + \beta_k a_k) \\ &= \beta_1^2 + \cdots + \beta_k^2 \\ &= \|\beta\|^2. \end{aligned}$$

(Going from the third to the fourth expression we use $a_i^T a_i = 1$, $a_i^T a_j = 0$ for $j \neq i$.) So we have the simple formula $\|x\| = \|\beta\|$.

5.5 *Orthogonalizing vectors.* Suppose that a and b are any n-vectors. Show that we can always find a scalar γ so that $(a - \gamma b) \perp b$, and that γ is unique if $b \neq 0$. (Give a formula for the scalar γ.) In other words, we can always subtract a multiple of a vector from another one, so that the result is orthogonal to the original vector. The orthogonalization step in the Gram–Schmidt algorithm is an application of this.

Solution. Two vectors are orthogonal if their inner product is zero. So we want to find γ such that $(a - \gamma b)^T b = a^T b - \gamma b^T b = 0$. If $b = 0$, then we can choose any γ, and we have $(a - \gamma b) \perp b$, since all vectors are orthogonal to 0. If $b \neq 0$, then $b^T b = \|b\|^2 \neq 0$, and we can take $\gamma = a^T b / b^T b$.

5.6 *Gram–Schmidt algorithm.* Consider the list of n n-vectors

$$a_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \qquad a_2 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \qquad \ldots, \qquad a_n = \begin{bmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}.$$

(The vector a_i has its first i entries equal to one, and the remaining entries zero.) Describe what happens when you run the Gram–Schmidt algorithm on this list of vectors, *i.e.*, say what $q_1, \ldots, q_n$ are. Is $a_1, \ldots, a_n$ a basis?

Solution. $\tilde{q}_1 = a_1$ has norm one, so we have $q_1 = a_1 = e_1$. $\tilde{q}_2 = a_2 - (q_1^T a_2)q_2 = e_2$ also has norm one, so $q_2 = e_2$. This pattern continues, and we see that $q_i = e_i$.

Yes, they are a basis. We know this since the Gram–Schmidt algorithm terminates after processing all n vectors. We can also see this directly. Suppose that $\beta_1 a_1 + \cdots + \beta_n a_n = 0$. The last entry of this vector is β_n, so we know that $\beta_n = 0$. This implies that $\beta_1 a_1 + \cdots + \beta_{n-1} a_{n-1} = 0$. The $(n-1)$ entry of this vector is β_{n-1}, so we know that $\beta_{n-1} = 0$. This continues, and we conclude that all β_i are zero. So the vectors are linearly independent, and therefore a basis.

5.7 *Running Gram–Schmidt algorithm twice.* We run the Gram–Schmidt algorithm once on a given set of vectors $a_1, \ldots, a_k$ (we assume this is successful), which gives the vectors $q_1, \ldots, q_k$. Then we run the Gram–Schmidt algorithm on the vectors $q_1, \ldots, q_k$, which produces the vectors $z_1, \ldots, z_k$. What can you say about $z_1, \ldots, z_k$?

Solution. Since the original list of vectors is orthogonal, the orthogonalization step in the Gram–Schmidt algorithm does nothing; we have $\tilde{q}_i = q_i$. Since the vectors q_i have norm one, the normalization step also does nothing: $z_i = q_i$. If you run Gram–Schmidt a second time, it does nothing.

5.8 *Early termination of Gram–Schmidt algorithm.* When the Gram–Schmidt algorithm is run on a particular list of 10 15-vectors, it terminates in iteration 5 (since $\tilde{q}_5 = 0$). Which of the following must be true?

(a) a_2, a_3, a_4 are linearly independent.

(b) a_1, a_2, a_5 are linearly dependent.

(c) a_1, a_2, a_3, a_4, a_5 are linearly dependent.

(d) a_4 is nonzero.

Solution.

(a) False. If this were true, the algorithm would have terminated in iteration 4.

(b) Could be true or false. If the algorithm terminates in iteration 5, then the collection a_1, a_2, a_3, a_4 is linearly independent and the collection a_1, a_2, a_3, a_4, a_5 is linearly dependent. It is possible that the smaller collection a_1, a_2, a_5 is linearly dependent.

(c) True.

(d) False. If this were true, the algorithm would have terminated in iteration 4.

5.9 *A particular computer can carry out the Gram–Schmidt algorithm on a list of $k = 1000$ n-vectors, with $n = 10000$, in around 2 seconds. About how long would you expect it to take to carry out the Gram–Schmidt algorithm with $\tilde{k} = 500$ $\tilde{n}$-vectors, with $\tilde{n} = 1000$?*

Solution. First we can find (or really, estimate) the speed of the computer, which we will call S flop/s. Gram–Schmidt requires $2nk^2$ flops. From $2 \approx S/(2kn^2)$, with $k = 10^3$ and $n = 10^4$, we find that $S \approx 10^{10}$ flop/s, which is 10 Gflop/s. Using this we can guess how long it will take to carry out Gram–Schmidt on the second problem, as $(2\tilde{n}\tilde{k}^2)/10^{10} = 0.05$ seconds.

Here's another way to get to the same answer. The time to carry out Gram–Schmidt scales linearly with n and quadratically by k. So if we reduce n by a factor of 10, we save a factor of 10. When we reduce k by a factor of two, we save an additional factor of 4. So with the new values of n and k (*i.e.*, $\tilde{n}$ and $\tilde{k}$), we should be able to carry out Gram–Schmidt $40\times$ faster, and $2/40 = 0.05$ seconds, as above.
(Verified on Boyd's laptop.)

Chapter 6

Matrices

Exercises

6.1 *Matrix and vector notation.* Suppose $a_1, \ldots, a_n$ are m-vectors. Determine whether each expression below makes sense (*i.e.*, uses valid notation). If the expression does make sense, give its dimensions.

(a) $\begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix}$

(b) $\begin{bmatrix} a_1^T \\ \vdots \\ a_n^T \end{bmatrix}$

(c) $\begin{bmatrix} a_1 & \cdots & a_n \end{bmatrix}$

(d) $\begin{bmatrix} a_1^T & \cdots & a_n^T \end{bmatrix}$

Solution.

(a) In this expression we are stacking n m-vectors, so the expression is an nm-vector. (Or $(nm) \times 1$ matrix.)

(b) In this expression we are stacking n $1 \times m$ matrices (*i.e.*, row vectors), so the expression is an $n \times m$ matrix.

(c) In this expression we are concatenating n m-vectors, so the expression is an $m \times n$ matrix.

(d) In this expression we are concatenating n m-row-vectors or $1 \times m$ matrices, so it is a $1 \times (mn)$ matrix or (mn)-row-vector.

6.2 *Matrix notation.* Suppose the block matrix

$$\begin{bmatrix} A & I \\ I & C \end{bmatrix}$$

makes sense, where A is a $p \times q$ matrix. What are the dimensions of C?

Solution. A has p rows, so the I in the upper right must also have p rows. But I is always square, so it has p columns as well. It follows that C has p columns. A has q columns, so the I at lower left has q columns, and therefore also q rows. It follows that C has q rows. So, C is $q \times p$.

6.3 *Block matrix.* Assuming the matrix

$$K = \begin{bmatrix} I & A^T \\ A & 0 \end{bmatrix}$$

makes sense, which of the following statements must be true? ('Must be true' means that it follows with no additional assumptions.)

(a) K is square.

(b) A is square or wide.

(c) K is symmetric, *i.e.*, $K^T = K$.

(d) The identity and zero submatrices in K have the same dimensions.

(e) The zero submatrix is square.

Solution. Let's suppose that A is $m \times n$. Then A^T is $n \times m$. It follows that the identity matrix I is $n \times n$, and the zero matrix 0 is $m \times m$. The matrix A can be any dimensions.

(a) Must be true. K is $(m+n) \times (m+n)$.

(b) Might not be true. A could be tall.

(c) Must be true. Using the block transpose formula on K, you get K back again.

(d) Might not be true. They are $n \times n$ and $m \times m$, respectively, and we don't need to have $m = n$.

(e) Must be true. The zero submatrix has dimensions $m \times m$.

6.4 *Adjacency matrix row and column sums.* Suppose A is the adjacency matrix of a directed graph (see page 112). What are the entries of the vector $A\mathbf{1}$? What are the entries of the vector $A^T\mathbf{1}$?

Solution. $A_{ij} = 1$ means an edge goes from node j to node i. When we sum over j, we get the total number of edges that go into node i. This is called the *in-degree* of node i. So the ith entry of the vector $A\mathbf{1}$ is the in-degree of node i.

The vector $A^T\mathbf{1}$ sums the columns of A. The ith entry is the number of edges that start node i. This number is called the out-degree of the nodes.

6.5 *Adjacency matrix of reversed graph.* Suppose A is the adjacency matrix of a directed graph (see page 112). The *reversed graph* is obtained by reversing the directions of all the edges of the original graph. What is the adjacency matrix of the reversed graph? (Express your answer in terms of A.)

Solution. The adjacency matrix of the reversed graph is A^T. To see this, we note that $A_{ij} = 1$ means there is an edge in the original graph from node j to node i. In the reversed graph, that edge goes from node i to node j, so its adjacency matrix $\tilde{A}$ satisfies $\tilde{A}_{ji} = 1$. This argument goes both ways, so $\tilde{A} = A^T$.

6.6 *Matrix-vector multiplication.* For each of the following matrices, describe in words how x and $y = Ax$ are related. In each case x and y are n-vectors, with $n = 3k$.

(a) $A = \begin{bmatrix} 0 & 0 & I_k \\ 0 & I_k & 0 \\ I_k & 0 & 0 \end{bmatrix}$.

(b) $A = \begin{bmatrix} E & 0 & 0 \\ 0 & E & 0 \\ 0 & 0 & E \end{bmatrix}$, where E is the $k \times k$ matrix with all entries $1/k$.

Solution.

(a) y is x with the first k entries swapped with the last k entries.

(b) y is x with the first, middle, and last groups of k entries replaced with their group averages.

6.7 *Currency exchange matrix.* We consider a set of n currencies, labeled $1, \ldots, n$. (These might correspond to USD, RMB, EUR, and so on.) At a particular time the exchange or conversion rates among the n currencies are given by an $n \times n$ (exchange rate) matrix R, where R_{ij} is the amount of currency i that you can buy for one unit of currency j. (All entries of R are positive.) The exchange rates include commission charges, so we have $R_{ji}R_{ij} < 1$ for all $i \neq j$. You can assume that $R_{ii} = 1$.

Suppose $y = Rx$, where x is a vector (with nonnegative entries) that represents the amounts of the currencies that we hold. What is y_i? Your answer should be in English.

Solution. y_i is the amount of currency i we can obtain by converting all our currencies into currency i. To see this, we note that the equation for y_i is

$$y_i = R_{i1}x_1 + \cdots + R_{in}x_n.$$

$R_{ij}x_j$ is the amount of of currency i we would get if we converted our holdings of currency j, x_j, into currency i. Summing over j we see that y_i is total of currency i we would get if we converted each of our currencies into currency i.

6.8 *Cash flow to bank account balance.* The T-vector c represents the cash flow for an interest bearing bank account over T time periods. Positive values of c indicate a deposit, and negative values indicate a withdrawal. The T-vector b denotes the bank account balance in the T periods. We have $b_1 = c_1$ (the initial deposit or withdrawal) and

$$b_t = (1+r)b_{t-1} + c_t, \quad t = 2, \ldots, T,$$

where $r > 0$ is the (per-period) interest rate. (The first term is the previous balance plus the interest, and the second term is the deposit or withdrawal.)

Find the $T \times T$ matrix A for which $b = Ac$. That is, the matrix A maps a cash flow sequence into a bank account balance sequence. Your description must make clear what all entries of A are.

Solution. We apply the expression for b_t recursively to find b as a function of c:

$$\begin{aligned}
b_1 &= c_1 \\
b_2 &= (1+r)b_1 + c_2 \\
&= (1+r)c_1 + c_2 \\
b_3 &= (1+r)b_2 + c_3 \\
&= (1+r)^2 c_1 + (1+r)c_2 + c_3 \\
&\vdots \\
b_T &= (1+r)b_{T-1} + c_T \\
&= (1+r)^{T-1} c_1 + (1+r)^{T-2} c_2 + \cdots + (1+r)c_{T-1} + c_T.
\end{aligned}$$

This can be written as $b = Ac$, where A is the lower triangular matrix

$$A = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1+r & 1 & 0 & \cdots & 0 \\ (1+r)^2 & 1+r & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (1+r)^{T-1} & (1+r)^{T-2} & (1+r)^{T-3} & \cdots & 1 \end{bmatrix}.$$

The elements of A are

$$A_{ij} = \begin{cases} (1+r)^{i-j} & i \geq j \\ 0 & i < j. \end{cases}$$

6.9 *Multiple channel marketing campaign.* Potential customers are divided into m market segments, which are groups of customers with similar demographics, *e.g.*, college educated women aged 25–29. A company markets its products by purchasing advertising in a set of n channels, *i.e.*, specific TV or radio shows, magazines, web sites, blogs, direct mail, and so on. The ability of each channel to deliver impressions or views by potential customers is characterized by the *reach matrix*, the $m \times n$ matrix R, where R_{ij} is the number of views of customers in segment i for each dollar spent on channel j. (We assume that the total number of views in each market segment is the sum of the views from each channel, and that the views from each channel scale linearly with spending.) The n-vector c will denote the company's purchases of advertising, in dollars, in the n channels. The m-vector v gives the total number of impressions in the m market segments due to the advertising in all channels. Finally, we introduce the m-vector a, where a_i gives the profit in dollars per impression in market segment i. The entries of R, c, v, and a are all nonnegative.

(a) Express the total amount of money the company spends on advertising using vector/matrix notation.

(b) Express v using vector/matrix notation, in terms of the other vectors and matrices.

(c) Express the total profit from all market segments using vector/matrix notation.

(d) How would you find the single channel most effective at reaching market segment 3, in terms of impressions per dollar spent?

(e) What does it mean if R_{35} is very small (compared to other entries of R)?

Solution.

(a) $\mathbf{1}^T c$. The company's purchases in advertising for the n channels is given by c, so summing up the entries of c gives the total amount of money the company spends across all channels.

(b) $v = Rc$. To find the total number of impressions v_i in the ith market segment, we need to sum up the impressions from each channel, which is given by an inner product of the ith row of R and the amounts c spent on advertising per channel.

(c) $a^T v = a^T Rc$. The total profit is the sum of the profits from each market segment, which is the product of the number of impressions for that segment v_i and the profit per impression for that segment a_i. The total profit is the sum $\sum_i a_i v_i = a^T v$. Substituting $v = Rc$ gives $a^T v = a^T Rc$.

(d) $\operatorname{argmax}_j R_{3j}$, *i.e.*, the column index of the largest entry in the third row of R. The number of impressions made on the third market segment, for each dollar spent, is given by the third row of R. The index of the greatest element in this row is then the channel that gives the highest number of impressions per dollar spent.

(e) The fifth channel makes relatively few impressions on the third market segment per dollar spent, compared to the other channels.

6.10 *Resource requirements.* We consider an application with n different job (types), each of which consumes m different resources. We define the $m \times n$ resource matrix R, with entry R_{ij} giving the amount of resource i that is needed to run one unit of job j, for $i = 1, \ldots, m$ and $j = 1, \ldots, n$. (These numbers are typically positive.) The number (or amount) of each of the different jobs to be processed or run is given by the entries of the n-vector x. (These entries are typically nonnegative integers, but they can be fractional if the jobs are divisible.) The entries of the m-vector p give the price per unit of each of the resources.

(a) Let y be the m-vector whose entries give the total of each of the m resources needed to process the jobs given by x. Express y in terms of R and x using matrix and vector notation.

(b) Let c be an n-vector whose entries gives the cost per unit for each job type. (This is the total cost of the resources required to run one unit of the job type.) Express c in terms of R and p using matrix and vector notation.

Remark. One example is a data center, which runs many instances of each of n types of application programs. The resources include number of cores, amount of memory, disk, and network bandwidth.

Solution. The required resource vector is given by $y = Rx$. The job cost vector is given by $c = R^T p$.

6.11 Let A and B be two $m \times n$ matrices. Under each of the assumptions below, determine whether $A = B$ must always hold, or whether $A = B$ holds only sometimes.

(a) Suppose $Ax = Bx$ holds for all n-vectors x.

(b) Suppose $Ax = Bx$ for some nonzero n-vector x.

Solution.

(a) $A = B$ *always.* Since $Ae_i = Be_i$ for each i, we see that each column of A equals the corresponding column of B, and that means $A = B$.

(b) $A = B$ *sometimes.* For example,

$$\begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 5 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix},$$

but the matrices are not equal.

6.12 *Skew-symmetric matrices.* An $n \times n$ matrix A is called *skew-symmetric* if $A^T = -A$, *i.e.*, its transpose is its negative. (A symmetric matrix satisfies $A^T = A$.)

(a) Find all 2×2 skew-symmetric matrices.

(b) Explain why the diagonal entries of a skew-symmetric matrix must be zero.

(c) Show that for a skew-symmetric matrix A, and any n-vector x, $(Ax) \perp x$. This means that Ax and x are orthogonal. *Hint.* First show that for any $n \times n$ matrix A and n-vector x, $x^T(Ax) = \sum_{i,j=1}^n A_{ij}x_ix_j$.

(d) Now suppose A is any matrix for which $(Ax) \perp x$ for any n-vector x. Show that A must be skew-symmetric. *Hint.* You might find the formula

$$(e_i + e_j)^T(A(e_i + e_j)) = A_{ii} + A_{jj} + A_{ij} + A_{ji},$$

valid for any $n \times n$ matrix A, useful. For $i = j$, this reduces to $e_i^T(Ae_i) = A_{ii}$.

Solution.

(a) The equality $A^T = -A$ for a 2×2 matrix means

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}^T = \begin{bmatrix} A_{11} & A_{21} \\ A_{12} & A_{22} \end{bmatrix} = \begin{bmatrix} -A_{11} & -A_{12} \\ -A_{21} & -A_{22} \end{bmatrix}.$$

From $A_{11} = -A_{11}$ we find that $A_{11} = 0$; similarly we get $A_{22} = 0$. We also have $A_{21} = -A_{12}$ and $A_{12} = -A_{21}$. These are the same equation; we can have any value of A_{12}, say α, and then we have

$$A = \alpha \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

as the general form of a skew-symmetric 2×2 matrix.

(b) Since $A^T = -A$, we have $(A^T)_{ii} = (-A)_{ii}$. But the left-hand side is A_{ii}, so we have $A_{ii} = -A_{ii}$. This implies that $A_{ii} = 0$.

(c) First we establish the formula given in follow the hint:

$$x^T(Ax) = \sum_{i=1}^n x_i(Ax)_i = \sum_{i=1}^n x_i \sum_{j=1}^n A_{ij}x_j = \sum_{i,j=1}^n A_{ij}x_ix_j.$$

Now suppose that A is skew-symmetric. This means that (by part (b)) $A_{ii} = 0$. Also, $A_{ij} = -A_{ji}$. In the sum above, we lump the term $A_{ij}x_ix_j$ and $A_{ji}x_jx_i$ together, and we see that they cancel each other, *i.e.*, add to zero. So when we sum over all i and j, we get zero, and it follows that $x^T(Ax) = 0$, which means that $(Ax) \perp x$.

(d) For $x = e_i$, the property $(Ax) \perp x$ gives $A_{ii} = e_i^T Ae_i = 0$. All the diagonal entries are zero. Applying the same property with $x = e_i + e_j$ and using the expression in the hint then gives

$$0 = (e_i + e_j)^T A(e_i + e_j) = A_{ij} + A_{ji}.$$

We conclude that if $(Ax) \perp x$ for all x, then $A_{ij} = -A_{ji}$ for all i and j. In other words A is skew-symmetric.

6.13 *Polynomial differentiation.* Suppose p is a polynomial of degree $n-1$ or less, given by $p(t) = c_1 + c_2 t + \cdots + c_n t^{n-1}$. Its derivative (with respect to t) $p'(t)$ is a polynomial of degree $n-2$ or less, given by $p'(t) = d_1 + d_2 t + \cdots + d_{n-1} t^{n-2}$. Find a matrix D for which $d = Dc$. (Give the entries of D, and be sure to specify its dimensions.)

Solution. The derivative is the polynomial

$$p'(t) = c_2 + 2c_3 t + \cdots + c_n(n-1)t^{n-2},$$

so

$$d_1 = c_2, \qquad d_2 = 2c_3, \qquad d_3 = 3c_4, \quad \ldots \quad d_{n-1} = (n-1)c_n.$$

Since c is an n-vector and d is an $(n-1)$-vector, the matrix D must be $(n-1) \times n$. It is given by

$$D = \begin{bmatrix} 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 2 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & n-1 \end{bmatrix} = \begin{bmatrix} 0_{(n-1)\times 1} & \mathbf{diag}(1, 2, \ldots, n-1) \end{bmatrix}.$$

6.14 *Norm of matrix-vector product.* Suppose A is an $m \times n$ matrix and x is an n-vector. A famous inequality relates $\|x\|$, $\|A\|$, and $\|Ax\|$:

$$\|Ax\| \leq \|A\|\|x\|.$$

The left-hand side is the (vector) norm of the matrix-vector product; the right-hand side is the (scalar) product of the matrix and vector norms. Show this inequality. *Hints.* Let a_i^T be the ith row of A. Use the Cauchy–Schwarz inequality to get $(a_i^T x)^2 \leq \|a_i\|^2\|x\|^2$. Then add the resulting m inequalities.

Solution. Following the hint we get

$$\|Ax\|^2 = \sum_{i=1}^{m} (a_i^T x)^2 \leq \sum_{i=1}^{m} \|a_i\|^2\|x\|^2 = \|A\|^2\|x\|^2,$$

using the fact that the sum of the squared norms of the rows of a matrix is the squared norm of the matrix. Taking the squareroot gives the inequality.

6.15 *Distance between adjacency matrices.* Let A and B be the $n \times n$ adjacency matrices of two directed graphs with n vertices (see page 112). The squared distance $\|A - B\|^2$ can be used to express how different the two graphs are. Show that $\|A - B\|^2$ is the total number of directed edges that are in one of the two graphs but not in the other.

Solution Consider the i, j term in the sum

$$\|A - B\|^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} (A_{ij} - B_{ij})^2.$$

There are three cases

$$(A_{ij} - B_{ij})^2 = \begin{cases} 0 & \text{there is an edge from } j \text{ to } i \text{ in both graphs} \\ 0 & \text{there is no edge from } j \text{ to } i \text{ in both graphs} \\ 1 & \text{there is an edge from } j \text{ to } i \text{ in one the two graphs but not the other.} \end{cases}$$

Therefore the sum is equal to the number of directed edges that are present in one of the graphs but not in the other.

6.16 *Columns of difference matrix.* Are the columns of the difference matrix D, defined in (6.5), linearly independent?

Solution. We let $d_1, \ldots, d_n$ denote the columns of D. The columns $d_1, \ldots, d_n$ are linearly dependent. This follows instantly from the dimension-independence inequality. In fact, it's easy to find some coefficients: Take $\beta_1 = \cdots = \beta_n = 1$. If you add up all the columns, you'll see that in each row of D there are two contributions: one -1 and one $+1$, so the result is zero.

6.17 *Stacked matrix.* Let A be an $m \times n$ matrix, and consider the stacked matrix S defined by

$$S = \begin{bmatrix} A \\ I \end{bmatrix}.$$

When does S have linearly independent columns? When does S have linearly independent rows? Your answer can depend on m, n, or whether or not A has linearly independent columns or rows.

Solution.

(a) S *always* has linearly independent columns. To see this, suppose that $Sx = 0$. Since $Sx = (Ax, x)$, this means that $x = 0$.

(b) S *never* has linearly independent rows. S has $m+n$ rows, and each row has dimension n, so by the independence-dimension inequality, the rows are dependent.

6.18 *Vandermonde matrices.* A Vandermonde matrix is an $m \times n$ matrix of the form

$$V = \begin{bmatrix} 1 & t_1 & t_1^2 & \cdots & t_1^{n-1} \\ 1 & t_2 & t_2^2 & \cdots & t_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & t_m & t_m^2 & \cdots & t_m^{n-1} \end{bmatrix}$$

where $t_1, \ldots, t_m$ are numbers. Multiplying an n-vector c by the Vandermonde matrix V is the same as evaluating the polynomial of degree less than n, with coefficients $c_1, \ldots, c_n$, at the points $t_1, \ldots, t_m$; see page 120. Show that the columns of a Vandermonde matrix are linearly independent if the numbers $t_1, \ldots, t_m$ are distinct, *i.e.*, different from each other. *Hint.* Use the following fact from algebra: If a polynomial p with degree less than n has n or more roots (points t for which $p(t) = 0$) then all its coefficients are zero.

Solution. The columns of an $m \times n$ Vandermonde matrix V are linearly dependent if there exists an n-vector c, with not all entries 0, such that

$$Vc = \sum_{i=1}^{n} c_i v_i = 0.$$

The entries of the vector Vc are the values of the polynomial $p(t) = c_1 + c_2 t + \cdots + c_n t^{n-1}$ at $t_1, \ldots, t_m$:

$$Vc = \begin{bmatrix} c_1 + c_2 t_1 + c_3 t_1^2 + \cdots + c_n t_1^{n-1} \\ c_1 + c_2 t_2 + c_3 t_2^2 + \cdots + c_n t_2^{n-1} \\ \vdots \\ c_1 + c_2 t_m + c_3 t_m^2 + \cdots + c_n t_m^{n-1} \end{bmatrix} = \begin{bmatrix} p(t_1) \\ p(t_2) \\ \vdots \\ p(t_n) \end{bmatrix}.$$

If $Vc = 0$, then $p(t_i) = 0$ for $i = 1, \ldots, m$, so $p(t)$ has at least m distinct roots $t_1, \ldots, t_m$. Using the fact from algebra mentioned in the hint this is only possible if the coefficients of p are zero, *i.e.*, $c = 0$. We have shown that $Vc = 0$ implies $c = 0$, which means that the columns of V are linearly independent.

6.19 *Back-test timing.* The $T \times n$ asset returns matrix R gives the returns of n assets over T periods. (See page 120.) When the n-vector w gives a set of portfolio weights, the T-vector Rw gives the time series of portfolio return over the T time periods. Evaluating portfolio return with past returns data is called *back-testing*.

Consider a specific case with $n = 5000$ assets, and $T = 2500$ returns. (This is 10 years of daily returns, since there are around 250 trading days in each year.) About how long would it take to carry out this back-test, on a 1 Gflop/s computer?

Solution. A matrix-vector product Rw of an $T \times n$ matrix with an n-vector w takes $2Tn$ flops. For $n = 5000$ and $T = 2500$ this is $2.5\,10^7$ flops. On a 1 Gflop/s computer it takes about 25 milliseconds.

6.20 *Complexity of matrix-vector multiplication.* On page 123 we worked out the complexity of computing the m-vector Ax, where A is an $m \times n$ matrix and x is an n-vector, when each entry of Ax is computed as an inner product of a row of A and the vector x. Suppose instead that we compute Ax as a linear combination of the columns of A, with coefficients $x_1, \ldots, x_n$. How many flops does this method require? How does it compare to the method described on page 123?

Solution. First we compute the m-vectors $x_i a_i$, for $i = 1, \ldots, n$. This costs mn multiplies. Then we add these vectors, which requires $m(n-1)$ additions. So the total is $mn + m(n-1) = m(2n-1)$ flops, which is identical to the flop count using the other method.

6.21 *Complexity of matrix-sparse-vector multiplication.* On page 123 we consider the complexity of computing Ax, where A is a sparse $m \times n$ matrix and x is an n-vector x (not assumed to be sparse). Now consider the complexity of computing Ax when the $m \times n$ matrix A is not sparse, but the n-vector x is sparse, with $\mathbf{nnz}(x)$ nonzero entries. Give the total number of flops in terms of m, n, and $\mathbf{nnz}(x)$, and simplify it by dropping terms that are dominated by others when the dimensions are large. *Hint.* The vector Ax is a linear combination of $\mathbf{nnz}(x)$ columns of A.

Solution. Following the hint, we can form Ax as a linear combination of the columns of A that correspond to nonzeros in x. This is essentially multiplying a non sparse $m \times \mathbf{nnz}(x)$ matrix by x, which requires $m(2\,\mathbf{nnz}(x) - 1)$ flops. (Note that this does not depend on n at all.) We simplify this to $2m\,\mathbf{nnz}(x)$ flops.

6.22 *Distribute or not?* Suppose you need to compute $z = (A+B)(x+y)$, where A and B are $m \times n$ matrices and x and y are n-vectors.

(a) What is the approximate flop count if you evaluate z as expressed, *i.e.*, by adding A and B, adding x and y, and then carrying out the matrix-vector multiplication?

(b) What is the approximate flop count if you evaluate z as $z = Ax + Ay + Bx + By$, *i.e.*, with four matrix-vector multiplies and three vector additions?

(c) Which method requires fewer flops? Your answer can depend on m and n. *Remark.* When comparing two computation methods, we usually do not consider a factor of 2 or 3 in flop counts to be significant, but in this exercise you can.

Solution.

(a) Adding A and B costs mn flops, adding x and y costs n, and multiplying $(A+B)(x+y)$ costs $2mn$ flops, so the total is $3mn+n$ flops. This is approximately $3mn$ flops.

(b) Each matrix-vector multiply costs $2mn$ flops, so the total is $8mn$ flops. The three vector adds cost $3m$ flops. The total is $8mn + 3m$ flops, but we can approximate this as $8mn$ flops.

(c) The first method always requires fewer flops.

Chapter 7

Matrix examples

Exercises

7.1 *Projection on a line.* Let $P(x)$ denote the projection of the 2-D point (2-vector) x onto the line that passes through $(0,0)$ and $(1,3)$. (This means that $P(x)$ is the point on the line that is closest to x; see exercise 3.12.) Show that P is a linear function, and give the matrix A for which $P(x) = Ax$ for any x.

Solution To find $P(x)$ we minimize the distance between the point (x_1, x_2) and a point $(t, 3t)$ on the line through $(1,3)$ and the origin. The squared distance is a quadratic function

$$(t - x_1)^2 + (3t - x_2)^2 = 10t^2 - (2x_1 + 6x_2)t + x_1^2 + x_2^2.$$

Setting the derivative equal to zero gives $t = (x_1 + 3x_2)/10$, so the projection is

$$P(x) = \begin{bmatrix} t \\ 3t \end{bmatrix} = \begin{bmatrix} (x_1 + 3x_2)/10 \\ 3(x_1 + 3x_2)/10) \end{bmatrix}.$$

The function $P(x)$ is linear because it can be written as $P(x) = Ax$ with

$$A = \begin{bmatrix} 1/10 & 3/10 \\ 3/10 & 9/10 \end{bmatrix}.$$

In exercise 3.12 we considered the general problem of projecting an n-vector x on the line through two n-vectors a and b. We found the expression

$$P(x) = a + \frac{(x-a)^T(b-a)}{\|b-a\|^2}(b-a)$$

for the projection of x on the line. Applying this with $a = (0,0)$, $b = (1,3)$ gives the same expression as above. For general a, b the function $P(x)$ is affine. It can be written as

$$\begin{aligned} P(x) &= a - \frac{a^T(b-a)}{\|b-a\|^2}(b-a) + \frac{x^T(b-a)}{\|b-a\|^2}(b-a) \\ &= a - \frac{a^T(b-a)}{\|b-a\|^2}(b-a) + \frac{1}{\|b-a\|^2}(b-a)(x^T(b-a)) \\ &= a - \frac{a^T(b-a)}{\|b-a\|^2}(b-a) + \frac{1}{\|b-a\|^2}(b-a)(b-a)^T x \end{aligned}$$

and this has the form $P(x) = Cx + d$ with

$$C = \frac{1}{\|b-a\|^2}(b-a)(b-a)^T, \qquad d = a - \frac{a^T(b-a)}{\|b-a\|^2}(b-a).$$

The function is linear ($d = 0$) if the line through a and b passes through the origin, *i.e.*, if either $a = 0$, or a is nonzero and b is a scalar multiple of a. This is clear if $a = 0$. If $a \neq 0$ and $b = \lambda a$, then

$$d = a - \frac{a^T(\lambda a - a)}{\|\lambda a - a\|^2}(\lambda a - a) = a - \frac{(\lambda - 1)\|a\|^2}{|\lambda - 1|^2\|a\|^2}(\lambda - 1)a = 0.$$

7.2 *3-D rotation.* Let x and y be 3-vectors representing positions in 3-D. Suppose that the vector y is obtained by rotating the vector x about the vertical axis (*i.e.*, e_3) by 45° (counterclockwise, *i.e.*, from e_1 toward e_2). Find the 3×3 matrix A for which $y = Ax$. *Hint.* Determine the three columns of A by finding the result of the transformation on the unit vectors e_1, e_2, e_3.

Solution. We will construct A column by column. If we apply the given geometric transformation to e_1, we obtain the result $(1/\sqrt{2}, 1/\sqrt{2}, 0)$. This is the first column of A. When we apply the transformation to e_2, we get the result $(-1/\sqrt{2}, 1/\sqrt{2}, 0)$, which is the second column of A. Finally, when we apply the transformation to e_3, we simply get $e_3 = (0, 0, 1)$, the third column of A. So we have

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

7.3 *Trimming a vector.* Find a matrix A for which $Ax = (x_2, \ldots, x_{n-1})$, where x is an n-vector. (Be sure to specify the size of A, and describe all its entries.)

Solution. A is an $(n-2) \times n$ matrix with entries

$$A_{ij} = \begin{cases} 1 & i = j - 1 \\ 0 & \text{otherwise.} \end{cases}$$

Written out in terms of its entries, we have

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0_{(n-2)\times 1} & I_{(n-2)} & 0_{(n-2)\times 1} \end{bmatrix}.$$

The subscripts in the last expression indicate the dimensions for clarity.

We can justify this choice of A several ways. Working by columns, we can ask what should Ae_1 be? The answer is 0. This is the first column of A. What is Ae_2? It is e_1, which is the second column of A. And so on.

We can also work with the rows. The first row of A tells us how to get $(Ax)_1$. This should by choosing x_2, which means the first row is e_2^T. And so on.

7.4 *Down-sampling and up-conversion.* We consider n-vectors x that represent signals, with x_k the value of the signal at time k for $k = 1, \ldots, n$. Below we describe two functions of x that produce new signals $f(x)$. For each function, give a matrix A such that $f(x) = Ax$ for all x.

(a) $2\times$ *downsampling.* We assume n is even and define $f(x)$ as the $n/2$-vector y with elements $y_k = x_{2k}$. To simplify your notation you can assume that $n = 8$, *i.e.*,

$$f(x) = (x_2,\ x_4,\ x_6,\ x_8).$$

(On page 131 we describe a different type of down-sampling, that uses the average of pairs of original values.)

(b) $2\times$ *up-conversion with linear interpolation.* We define $f(x)$ as the $(2n-1)$-vector y with elements $y_k = x_{(k+1)/2}$ if k is odd and $y_k = (x_{k/2} + x_{k/2+1})/2$ if k is even. To simplify your notation you can assume that $n = 5$, *i.e.*,

$$f(x) = \left(x_1,\ \frac{x_1 + x_2}{2},\ x_2,\ \frac{x_2 + x_3}{2},\ x_3,\ \frac{x_3 + x_4}{2},\ x_4,\ \frac{x_4 + x_5}{2},\ x_5\right).$$

Solution

(a) The matrix A has size $n/2 \times n$ and its elements are

$$a_{ij} = \begin{cases} 1 & j = 2i \\ 0 & \text{otherwise.} \end{cases}$$

For example, if $n = 8$,

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

(b) The matrix has size $(2n-1) \times n$ and its elements are

$$a_{ij} = \begin{cases} 1 & i \text{ is odd and } j = (i+1)/2 \\ 1/2 & i \text{ is even and } j = i/2 \text{ or } j = i/2+1 \\ 0 & \text{otherwise.} \end{cases}$$

For example, if $n = 5$,

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

7.5 *Transpose of selector matrix.* Suppose the $m \times n$ matrix A is a selector matrix. Describe the relation between the m-vector u and the n-vector $v = A^T u$.

Solution. The product is

$$v = \begin{bmatrix} e_{k_1} & e_{k_2} & \cdots & e_{k_m} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{bmatrix} = u_1 e_{k_1} + u_2 e_{k_2} + \cdots + u_m e_{k_m}.$$

The entry v_j is the sum of the entries of u_i for which $k_i = 1$.

7.6 *Rows of incidence matrix.* Show that the rows of the incidence matrix of a graph are always linearly dependent. *Hint.* Consider the sum of the rows.

Solution. The sum of the rows is zero, *i.e.*, $\mathbf{1}^T A = 0$.

7.7 *Incidence matrix of reversed graph.* (See exercise 6.5.) Suppose A is the incidence matrix of a graph. The reversed graph is obtained by reversing the directions of all the edges of the original graph. What is the incidence matrix of the reversed graph? (Express your answer in terms of A.)

Solution. $-A$.

7.8 *Flow conservation with sources.* Suppose that A is the incidence matrix of a graph, x is the vector of edge flows, and s is the external source vector, as described in §7.3. Assuming that flow is conserved, *i.e.*, $Ax + s = 0$, show that $\mathbf{1}^T s = 0$. This means that the total amount injected into the network by the sources ($s_i > 0$) must exactly balance the total amount removed from the network at the sink nodes ($s_i < 0$). For example if the network is a (lossless) electrical power grid, the total amount of electrical power generated (and injected into the grid) must exactly balance the total electrical power consumed (from the grid).

Solution. Multiply $0 = Ax + s$ on the left by $\mathbf{1}^T$ to get

$$0 = \mathbf{1}^T(Ax + s) = (\mathbf{1}^T A)x + \mathbf{1}^T s = \mathbf{1}^T s.$$

$\mathbf{1}^T A = 0$ because $\mathbf{1}^T A$ is the row vector of columns sums of A, which are all exactly zero since each column has one $+1$ and one -1. So we get $\mathbf{1}^T s = 0$.

7.9 *Social network graph.* Consider a group of n people or users, and some symmetric social relation among them. This means that some pairs of users are *connected*, or *friends* (say). We can create a directed graph by associating a node with each user, and an edge between each pair of friends, arbitrarily choosing the direction of the edge. Now consider an n-vector v, where v_i is some quantity for user i, for example, age or education level (say, given in years). Let $\mathcal{D}(v)$ denote the Dirichlet energy associated with the graph and v, thought of as a potential on the nodes.

(a) Explain why the number $\mathcal{D}(v)$ does not depend on the choice of directions for the edges of the graph.

(b) Would you guess that $\mathcal{D}(v)$ is small or large? This is an open-ended, vague question; there is no right answer. Just make a guess as to what you might expect, and give a short English justification of your guess.

Solution.

(a) The Dirichlet energy has the form

$$\mathcal{D}(v) = \sum_{(i,j)\in\mathcal{E}} (v_i - v_j)^2,$$

where $\mathcal{E}$ is the set of edges of the graph, which correspond to pairs (i, j) of individuals who are friends. If we switch the orientation of edge between i and j, the term $(v_i - v_j)^2$ becomes $(v_j - v_i)^2$, which is the same.

(b) You might guess that $\mathcal{D}(v)$ is small. This means that friends tend to have similar values of the quantity represented by v. For example, your circle of friends probably mostly have ages similar to yours, education levels similar to yours, and so on.

7.10 *Circle graph.* A *circle graph* (also called a *cycle graph*) has n vertices, with edges pointing from vertex 1 to vertex 2, from vertex 2 to vertex 3, ..., from vertex $n-1$ to vertex n, and finally, from vertex n to vertex 1. (This last edge completes the circle.)

(a) Draw a diagram of a circle graph, and give its incidence matrix A.

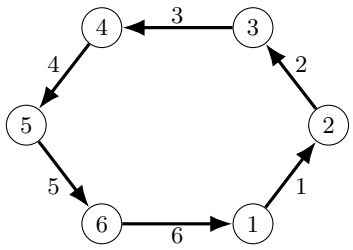
(b) Suppose that x is a circulation for a circle graph. What can you say about x?

(c) Suppose the n-vector v is a potential on a circle graph. What is the Dirichlet energy $\mathcal{D}(v) = \|A^T v\|^2$?

Remark. The circle graph arises when an n-vector v represents a periodic time series. For example, v_1 could be the value of some quantity on Monday, v_2 its value on Tuesday, and v_7 its value on Sunday. The Dirichlet energy is a measure of the roughness of such an n-vector v.

Solution.

(a) The figure shows the circle graph with 6 nodes.



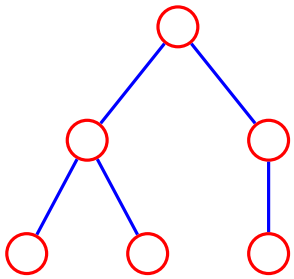


Figure 7.1 Tree with six vertices.

The incidence matrix is

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

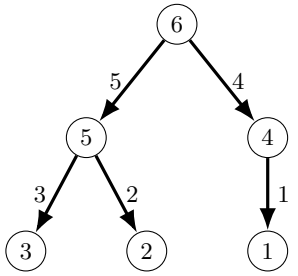
(b) The entries of x are equal.

(c) $\mathcal{D}(v) = (v_2 - v_1)^2 + (v_3 - v_2)^2 + \cdots + (v_n - v_{n-1})^2 + (v_1 - v_n)^2$.

7.11 *Tree.* An undirected graph is called a tree if it is connected (there is a path from every vertex to every other vertex) and contains no cycles, *i.e.*, there is no path that begins and ends at the same vertex. Figure 7.1 shows a tree with six vertices. For the tree in the figure, find a numbering of the vertices and edges, and an orientation of the edges, so that the incidence matrix A of the resulting directed graph satisfies $A_{ii} = 1$ for $i = 1, \ldots, 5$ and $A_{ij} = 0$ for $i < j$. In other words, the first 5 rows of A form a lower triangular matrix with ones on the diagonal.

Solution. For the numbering and orientation in the figure the incidence matrix is

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



7.12 *Some properties of convolution.* Suppose that a is an n-vector.

(a) *Convolution with* 1. What is $1 * a$? (Here we interpret 1 as a 1-vector.)

(b) *Convolution with a unit vector.* What is $e_k * a$, where e_k is the kth unit vector of dimension q? Describe this vector mathematically (*i.e.*, give its entries), and via a brief English description. You might find vector slice notation useful.

Solution.

(a) $1 * a = a$.

We apply the definition (7.2) with $m = 1$ and $b_1 = 1$. We sum over all pairs of indices i, j that satisfy $i + j = k + 1$. Since $j = 1$ is the only possible value for j, there is only one term ($j = 1$, $i = k$) and the sum is simply $c_k = a_k b_1 = a_k$.

(b) $e_k * a = (0_{k-1}, a, 0_{q-k})$.

Roughly speaking, $e_k * a$ is the vector a, shifted the right $k-1$ entries, and 'padded' on the end with $k - q$ additional zeros. To derive this we apply (7.2) with $m = q$ and $b = e_k$:

$$c_l = \sum_{i+j=l+1} a_i b_j, \quad l = 1, \ldots, n + q - 1.$$

The vector b has only one nonzero entry, $b_k = 1$. This entry appears in the sum only if $i = l + 1 - k$ is between 1 and n. In that case $c_l = a_{l+1-k}$. If $l + 1 - k$ is less than 1 or greater than n the sum is zero because b_j is zero in all terms. Therefore

$$c_l = \begin{cases} 0 & 1 \leq l \leq k-1 \\ a_{l+1-k} & k \leq l \leq n+k-1 \\ 0 & n+k \leq l \leq n+q-1. \end{cases}$$

7.13 *Sum property of convolution.* Show that for any vectors a and b, we have $\mathbf{1}^T(a * b) = (\mathbf{1}^T a)(\mathbf{1}^T b)$. In words: The sum of the coefficients of the convolution of two vectors is the product of the sums of the coefficients of the vectors. *Hint.* If the vector a represents the coefficients of a polynomial p, $\mathbf{1}^T a = p(1)$.

Solution. Let a and b be the vectors of coefficients of the polynomials

$$p(x) = a_1 + a_2 x + \cdots + a_m x^{m-1}, \qquad q(x) = b_1 + b_2 x + \cdots + b_n x^{n-1},$$

respectively. Then $c = a * b$ gives the coefficients of the product polynomial $r(x) = p(x)q(x)$:

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

Using the hint, we have

$$\mathbf{1}^T(a * b) = r(1) = p(1)q(1) = (\mathbf{1}^T a)(\mathbf{1}^T b).$$

This is also easily seen from the matrix representation of the convolution $c = T(a)b$ where $T(a)$ is the $(m + n - 1) \times n$ Toeplitz matrix with $(a_1, \ldots, a_m, 0, \ldots, 0)$ in its first column. We have

$$\mathbf{1}^T c = \mathbf{1}^T T(a) b = \begin{bmatrix} \mathbf{1}^T a & \mathbf{1}^T a & \cdots & \mathbf{1}^T a \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} = (\mathbf{1}^T a)(\mathbf{1}^T b).$$

In the second step we note that $\mathbf{1}^T T(a)$ is a row vector with ith element equal to the sum of the elements in column i of $T(a)$. These column sums are all equal to $\mathbf{1}^T a$.

7.14 *Rainfall and river height.* The T-vector r gives the daily rainfall in some region over a period of T days. The vector h gives the daily height of a river in the region (above its normal height). By careful modeling of water flow, or by fitting a model to past data, it is found that these vectors are (approximately) related by convolution: $h = g * r$, where

$$g = (0.1, 0.4, 0.5, 0.2).$$

Give a short story in English (with no mathematical terms) to approximately describe this relation. For example, you might mention how many days after a one day heavy rainfall the river height is most affected. Or how many days it takes for the river height to return to the normal height once the rain stops.

Solution. The first few entries of h are

$$\begin{aligned} h_1 &= 0.1r_1 \\ h_2 &= 0.4r_1 + 0.1r_2 \\ h_3 &= 0.5r_1 + 0.4r_2 + 0.1r_3 \\ h_4 &= 0.2r_1 + 0.5r_2 + 0.4r_3 + 0.1r_4 \\ h_5 &= 0.2r_2 + 0.5r_3 + 0.4r_4 + 0.1r_5 \\ h_6 &= 0.2r_3 + 0.5r_4 + 0.4r_5 + 0.1r_6 \end{aligned}$$

(assuming that $T \geq 6$). A one day rainfall has only a little effect on the river height on that day. Its main effect comes on the day after and the day after that. Its effect on the next day is smaller, and after 4 days it has no effect (if you believe this model).

7.15 *Channel equalization.* We suppose that $u_1, \ldots, u_m$ is a signal (time series) that is transmitted (for example by radio). A receiver receives the signal $y = c * u$, where the n-vector c is called the channel impulse response. (See page 138.) In most applications n is small, *e.g.*, under 10, and m is much larger. An *equalizer* is a k-vector h that satisfies $h * c \approx e_1$, the first unit vector of length $n + k - 1$. The receiver equalizes the received signal y by convolving it with the equalizer to obtain $z = h * y$.

(a) How are z (the equalized received signal) and u (the original transmitted signal) related? *Hint.* Recall that $h * (c * u) = (h * c) * u$.

(b) *Numerical example.* Generate a signal u of length $m = 50$, with each entry a random value that is either -1 or $+1$. Plot u and $y = c * u$, with $c = (1, 0.7, -0.3)$. Also plot the equalized signal $z = h * y$, with

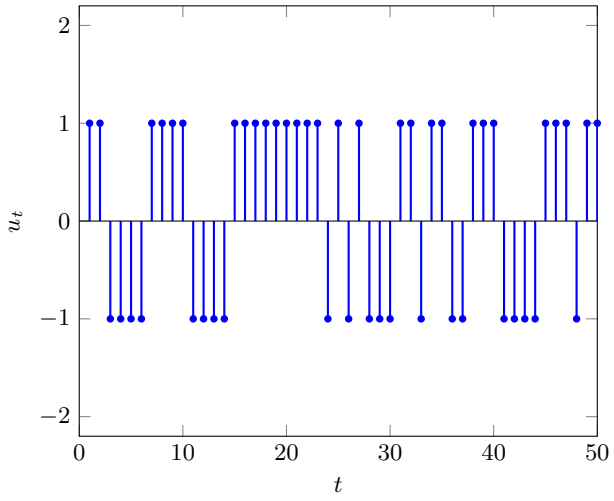
$$h = (0.9, -0.5, 0.5, -0.4, 0.3, -0.3, 0.2, -0.1).$$

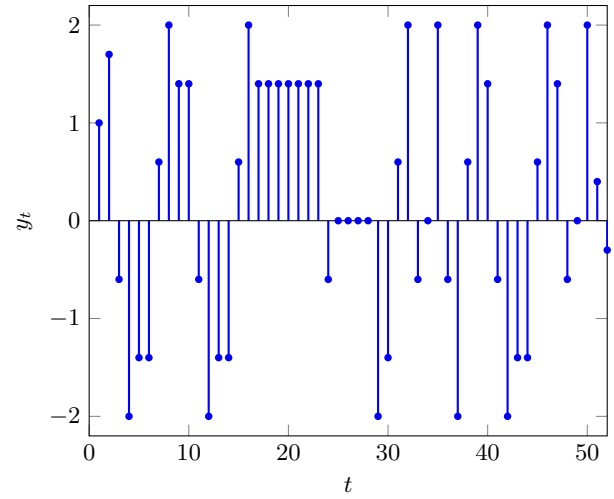
Solution.

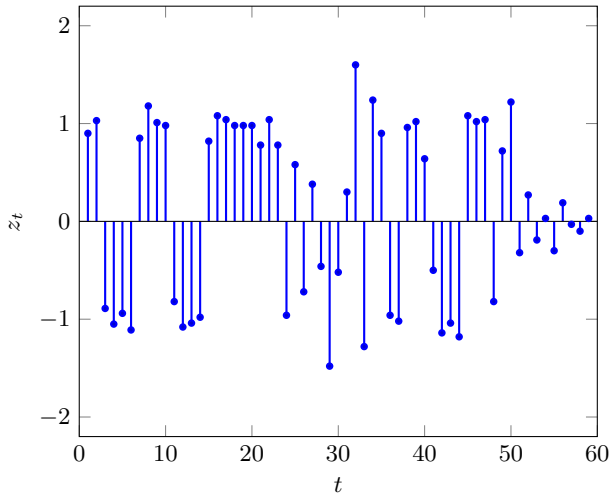
(a) The equalizer output is $z = h*(c*u) = (h*c)*u$. This is a vector of size $n+m+k-2$. Since $h * c \approx e_1$, we have

$$z \approx e_1 * u = (u_1, \ldots, u_m, 0, \ldots, 0).$$

(b) The figure shows u, y, z for a randomly chosen signal u. These three vectors have size 50, 52, and 59, respectively.







Chapter 8

Linear equations

Exercises

8.1 *Sum of linear functions.* Suppose $f : \mathbf{R}^n \to \mathbf{R}^m$ and $g : \mathbf{R}^n \to \mathbf{R}^m$ are linear functions. Their *sum* is the function $h : \mathbf{R}^n \to \mathbf{R}^m$, defined as $h(x) = f(x) + g(x)$ for any n-vector x. The sum function is often denoted as $h = f + g$. (This is another case of overloading the $+$ symbol, in this case to the sum of functions.) If f has matrix representation $f(x) = Fx$, and g has matrix representation $f(x) = Gx$, where F and G are $m \times n$ matrices, what is the matrix representation of the sum function $h = f + g$? Be sure to identify any $+$ symbols appearing in your justification.

Solution. The matrix $F + G$.

The function h can be expressed as

$$h(x) = f(x) + g(x) = Fx + Gx = (F + G)x.$$

The additions in $f(x) + g(x)$ and $Fx + Gx$ are additions of m-vectors. The addition in $F + G$ is addition of $m \times n$ matrices.

8.2 *Averages and affine functions.* Suppose that $G : \mathbf{R}^n \to \mathbf{R}^m$ is an affine function. Let $x_1, \ldots, x_k$ be n-vectors, and define the m-vectors $y_1 = G(x_1), \ldots, y_k = G(x_k)$. Let

$$\overline{x} = (x_1 + \cdots + x_k)/k, \qquad \overline{y} = (y_1 + \cdots + y_k)/k$$

be the averages of these two lists of vectors. (Here $\overline{x}$ is an n-vector and $\overline{y}$ is an m-vector.) Show that we always have $\overline{y} = G(\overline{x})$. In words: The average of an affine function applied to a list of vectors is the same as the affine function applied to the average of the list of vectors.

Solution. We can write $\overline{x}$ as

$$\overline{x} = \theta_1 x_1 + \cdots + \theta_k x_k,$$

where $\theta_1 = \cdots = \theta_k = 1/k$. Since these coefficients add up to one, we have (by superposition for affine functions)

$$\begin{aligned} G(\overline{x}) &= G(\theta_1 x_1 + \cdots + \theta_k x_k) \\ &= \theta_1 G(x_1) + \cdots + \theta_k G(x_k) \\ &= \theta_1 y_1 + \cdots + \theta_k y_k \\ &= \overline{y}. \end{aligned}$$

8.3 *Cross-product.* The cross product of two 3-vectors $a = (a_1, a_2, a_3)$ and $x = (x_1, x_2, x_3)$ is defined as the vector

$$a \times x = \begin{bmatrix} a_2 x_3 - a_3 x_2 \\ a_3 x_1 - a_1 x_3 \\ a_1 x_2 - a_2 x_1 \end{bmatrix}.$$

The cross product comes up in physics, for example in electricity and magnetism, and in dynamics of mechanical systems like robots or satellites. (You do not need to know this for this exercise.)

Assume a is fixed. Show that the function $f(x) = a \times x$ is a linear function of x, by giving a matrix A that satisfies $f(x) = Ax$ for all x.

Solution.

$$A = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}.$$

8.4 *Linear functions of images.* In this problem we consider several linear functions of a monochrome image with $N \times N$ pixels. To keep the matrices small enough to work out by hand, we will consider the case with $N = 3$ (which would hardly qualify as an image). We represent a 3×3 image as a 9-vector using the ordering of pixels shown below.

1	4	7
2	5	8
3	6	9

(This ordering is called *column-major*.) Each of the operations or transformations below defines a function $y = f(x)$, where the 9-vector x represents the original image, and the 9-vector y represents the resulting or transformed image. For each of these operations, give the 9×9 matrix A for which $y = Ax$.

(a) Turn the original image x upside-down.

(b) Rotate the original image x clockwise $90°$.

(c) Translate the image up by 1 pixel and to the right by 1 pixel. In the translated image, assign the value $y_i = 0$ to the pixels in the first column and the last row.

(d) Set each pixel value y_i to be the average of the neighbors of pixel i in the original image. By neighbors, we mean the pixels immediately above and below, and immediately to the left and right. The center pixel has 4 neighbors; corner pixels have 2 neighbors, and the remaining pixels have 3 neighbors.

Solution. In each subproblem, the vectors x and $y = f(x)$ contain the pixel intensities of an image and the transformed image, ordered using column-major ordering:

x_1	x_4	x_7
x_2	x_5	x_8
x_3	x_6	x_9

$\longrightarrow$

y_1	y_4	y_7
y_2	y_5	y_8
y_3	y_6	y_9

(a) The transformed image is

x_3	x_6	x_9
x_2	x_5	x_8
x_1	x_4	x_7

The function maps x to $f(x) = (x_3, x_2, x_1, x_6, x_5, x_4, x_9, x_8, x_7)$ and can be repre-

sented as $f(x) = Ax$, with

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

The matrix can be written more concisely as

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

where E is the 3×3 reverser matrix

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

(b) The transformed image is

x_7	x_8	x_9
x_4	x_5	x_6
x_1	x_2	x_3

The function maps x to $f(x) = (x_7, x_4, x_1, x_8, x_5, x_2, x_9, x_6, x_3)$ and can be represented as $f(x) = Ax$ with

$$\begin{aligned} A &= \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} e_7 & e_4 & e_1 & e_8 & e_5 & e_2 & e_9 & e_6 & e_3 \end{bmatrix} \end{aligned}$$

(c) The transformed image is

0	x_2	x_5
0	x_3	x_6
0	0	0

The function maps x to $f(x) = (0, 0, 0, x_2, x_3, 0, x_5, x_6, 0)$ and can be represented as $f(x) = Ax$ with

$$\begin{aligned} A &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & e_4 & e_5 & 0 & e_7 & e_8 & 0 & 0 & 0 \end{bmatrix} \end{aligned}$$

(d) The transformed image is

$\frac{1}{2}(x_2 + x_4)$	$\frac{1}{3}(x_1 + x_5 + x_7)$	$\frac{1}{2}(x_4 + x_8)$
$\frac{1}{3}(x_1 + x_3 + x_5)$	$\frac{1}{4}(x_2 + x_4 + x_6 + x_8)$	$\frac{1}{3}(x_5 + x_7 + x_9)$
$\frac{1}{2}(x_2 + x_6)$	$\frac{1}{3}(x_3 + x_5 + x_9)$	$\frac{1}{2}(x_6 + x_8)$

The function maps f to

$$\begin{aligned} f(x) &= (\frac{x_2 + x_4}{2}, \frac{x_1 + x_3 + x_5}{3}, \frac{x_2 + x_6}{2}, \frac{x_1 + x_5 + x_7}{3}, \frac{x_2 + x_4 + x_6 + x_8}{4}, \\ &\quad \frac{x_3 + x_5 + x_9}{3}, \frac{x_4 + x_8}{2}, \frac{x_5 + x_7 + x_9}{3}, \frac{x_6 + x_8}{2}) \end{aligned}$$

and can be represented as $f(x) = Ax$ with

$$A = \begin{bmatrix} 0 & 1/2 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 1/3 & 0 & 1/3 & 0 & 1/3 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 & 0 & 1/2 & 0 & 0 & 0 \\ 1/3 & 0 & 0 & 0 & 1/3 & 0 & 1/3 & 0 & 0 \\ 0 & 1/4 & 0 & 1/4 & 0 & 1/4 & 0 & 1/4 & 0 \\ 0 & 0 & 1/3 & 0 & 1/3 & 0 & 0 & 0 & 1/3 \\ 0 & 0 & 0 & 1/2 & 0 & 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 1/3 & 0 & 1/3 & 0 & 1/3 \\ 0 & 0 & 0 & 0 & 0 & 1/2 & 0 & 1/2 & 0 \end{bmatrix}.$$

8.5 *Symmetric and anti-symmetric part.* An n-vector x is *symmetric* if $x_k = x_{n-k+1}$ for $k = 1, \ldots, n$. It is *anti-symmetric* if $x_k = -x_{n-k+1}$ for $k = 1, \ldots, n$.

(a) Show that every vector x can be decomposed in a unique way as a sum $x = x_{\mathrm{s}} + x_{\mathrm{a}}$ of a symmetric vector x_{s} and an anti-symmetric vector x_{a}.

(b) Show that the symmetric and anti-symmetric parts $x_{\rm s}$ and $x_{\rm a}$ are linear functions of x. Give matrices $A_{\rm s}$ and $A_{\rm a}$ such that $x_{\rm s} = A_{\rm s}x$ and $x_{\rm a} = A_{\rm a}x$ for all x.

Solution.

(a) Suppose $x = u + v$ with u symmetric and v anti-symmetric. For $k = 1, \ldots, n$,

$$x_{n-k+1} = u_{n-k+1} + v_{n-k+1} = u_k - v_k.$$

Combining this with $x_k = u_k + v_k$, we find that the only possible values for u_k and v_k are

$$u_k = \frac{1}{2}(x_k + x_{n-k+1}), \qquad v_k = \frac{1}{2}(x_k - x_{n-k+1}).$$

These expressions define a symmetric and an anti-symmetric vector, since

$$\begin{aligned} u_{n-k+1} &= \frac{1}{2}(x_{n-k+1} + x_{n-(n-k+1)+1}) \\ &= \frac{1}{2}(x_{n-k+1} + x_k) \\ &= u_k, \\ v_{n-k+1} &= \frac{1}{2}(x_{n-k+1} - x_{n-(n-k+1)+1}) \\ &= \frac{1}{2}(x_{n-k+1} - x_k) \\ &= -v_k. \end{aligned}$$

(b) Let us denote by J_m the $m \times m$ reverser matrix:

$$J_m = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 1 & 0 \\ \vdots & \vdots & \iddots & \vdots & \vdots \\ 0 & 1 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \end{bmatrix}.$$

We distinguish two cases. If $n = 2m$ is even, then

$$A_{\rm s} = \begin{bmatrix} (1/2)I_m & (1/2)J_m \\ (1/2)J_m & (1/2)I_m \end{bmatrix}, \qquad A_{\rm a} = \begin{bmatrix} (1/2)I_m & -(1/2)J_m \\ -(1/2)J_m & (1/2)I_m \end{bmatrix}.$$

If $n = 2m + 1$ is odd, then

$$A_{\rm s} = \begin{bmatrix} (1/2)I_m & 0 & (1/2)J_m \\ 0 & 1 & 0 \\ (1/2)J_m & 0 & (1/2)I_m \end{bmatrix}, \qquad A_{\rm a} = \begin{bmatrix} (1/2)I_m & 0 & -(1/2)J_m \\ 0 & 0 & 0 \\ -(1/2)J_m & 0 & (1/2)I_m \end{bmatrix}.$$

8.6 *Linear functions.* For each description of y below, express it as $y = Ax$ for some A. (You should specify A.)

(a) y_i is the difference between x_i and the average of $x_1, \ldots, x_{i-1}$. (We take $y_1 = x_1$.)

(b) y_i is the difference between x_i and the average value of all other x_js, *i.e.*, the average of $x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$.

Solution.

(a) For $i > 1$, the ith entry of y is

$$y_i = x_i - \frac{1}{i-1}(x_1 + x_2 + \cdots + x_{i-1}).$$

A is lower triangular, with ones on the diagonal, and lower triangular entries $A_{ij} = -1/(i-1)$:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ -1 & 1 & 0 & 0 & \cdots & 0 \\ -1/2 & -1/2 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -1/(n-1) & -1/(n-1) & -1/(n-1) & -1/(n-1) & \cdots & 1 \end{bmatrix}.$$

(b) The ith entry of y is

$$y_i = x_i - \frac{1}{n-1}(x_1 + x_2 + \cdots + x_{i-1} + x_{i+1} + \cdots + x_n).$$

The diagonal entries of A are 1. The off-diagonal entries are all $-1/(n-1)$.

$$A = \begin{bmatrix} 1 & -1/(n-1) & \cdots & -1/(n-1) \\ -1/(n-1) & 1 & \cdots & -1/(n-1) \\ \vdots & \vdots & \ddots & \vdots \\ -1/(n-1) & -1/(n-1) & \cdots & 1 \end{bmatrix}$$

8.7 *Interpolation of polynomial values and derivatives.* The 5-vector c represents the coefficients of a quartic polynomial $p(x) = c_1 + c_2x + c_3x^2 + c_4x^3 + c_5x^4$. Express the conditions

$$p(0) = 0, \quad p'(0) = 0, \quad p(1) = 1, \quad p'(1) = 0,$$

as a set of linear equations of the form $Ac = b$. Is the system of equations underdetermined, over-determined, or square?

Solution. The four equations are

$$\begin{aligned} c_1 &= 0 \\ c_2 &= 0 \\ c_1 + c_2 + c_3 + c_4 + c_5 &= 1 \\ c_2 + 2c_3 + 3c_4 + 4c_5 &= 0. \end{aligned}$$

They can be written in the form $Ac = b$ as

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 & 4 \end{bmatrix}, \qquad b = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}.$$

The set of linear equations is underdetermined (4 equations in 5 variables).

8.8 *Interpolation of rational functions.* A *rational function* of degree two has the form

$$f(t) = \frac{c_1 + c_2t + c_3t^2}{1 + d_1t + d_2t^2},$$

where c_1, c_2, c_3, d_1, d_2 are coefficients. ('Rational' refers to the fact that f is a ratio of polynomials. Another name for f is *bi-quadratic*.) Consider the interpolation conditions

$$f(t_i) = y_i, \quad i = 1, \ldots, K,$$

where t_i and y_i are given numbers. Express the interpolation conditions as a set of linear equations in the vector of coefficients $\theta = (c_1, c_2, c_3, d_1, d_2)$, as $A\theta = b$. Give A and b, and their dimensions.

Solution. The interpolation conditions

$$\frac{c_1 + c_2 t_i + c_3 t_i^2}{1 + d_1 t_i + d_2 t_i^2} = y_i, \quad i = 1, \ldots, K,$$

can be written as linear equations

$$c_1 + c_2 t_i + c_3 t_i^2 - y_i d_i t_i - y_i d_2 t_i^2 = y_i, \quad i = 1, \ldots, K.$$

If we define $\theta = (c_1, c_2, c_3, d_1, d_2)$ then this can be written as $A\theta = b$ with

$$A = \begin{bmatrix} 1 & t_1 & t_1^2 & -y_1 t_1 & -y_1 t_1^2 \\ 1 & t_2 & t_2^2 & -y_2 t_2 & -y_2 t_2^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & t_K & t_K^2 & -y_K t_K & -y_K t_K^2 \end{bmatrix}, \qquad b = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_K \end{bmatrix}.$$

8.9 *Required nutrients.* We consider a set of n basic foods (such as rice, beans, apples) and a set of m nutrients or components (such as protein, fat, sugar, vitamin C). Food j has a cost given by c_j (say, in dollars per gram), and contains an amount N_{ij} of nutrient i (per gram). (The nutrients are given in some appropriate units, which can depend on the particular nutrient.) A daily diet is represented by an n-vector d, with d_i the daily intake (in grams) of food i. Express the condition that a diet d contains the total nutrient amounts given by the m-vector n^{des}, and has a total cost B (the budget) as a set of linear equations in the variables $d_1, \ldots, d_n$. (The entries of d must be nonnegative, but we ignore this issue here.)

Solution.

$$\begin{bmatrix} N_{11} & N_{12} & \cdots & N_{1n} \\ N_{21} & N_{22} & \cdots & N_{2n} \\ \vdots & \vdots & & \vdots \\ N_{m1} & N_{m2} & \cdots & N_{mn} \\ c_1 & c_2 & \cdots & c_n \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{bmatrix} = \begin{bmatrix} n_1^{\mathrm{des}} \\ n_2^{\mathrm{des}} \\ \vdots \\ n_m^{\mathrm{des}} \\ B \end{bmatrix}.$$

8.10 *Blending crude oil.* A set of K different types of crude oil are blended (mixed) together in proportions $\theta_1, \ldots, \theta_K$. These numbers sum to one; they must also be nonnegative, but we will ignore that requirement here. Associated with crude oil type k is an n-vector c_k that gives its concentration of n different constituents, such as specific hydrocarbons. Find a set of linear equations on the blending coefficients, $A\theta = b$, that expresses the requirement that the blended crude oil achieves a target set of constituent concentrations, given by the n-vector c^{tar}. (Include the condition that $\theta_1 + \cdots + \theta_K = 1$ in your equations.)

Solution.

$$\begin{bmatrix} c_1 & c_2 & \cdots & c_K \\ 1 & 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_K \end{bmatrix} = \begin{bmatrix} c^{\mathrm{tar}} \\ 1 \end{bmatrix}.$$

The matrix on the left-hand side has size $(n+1) \times K$.

8.11 *Location from range measurements.* The 3-vector x represents a location in 3-D. We measure the distances (also called the range) of x to four points at known locations a_1, a_2, a_3, a_4:

$$\rho_1 = \|x - a_1\|, \qquad \rho_2 = \|x - a_2\|, \qquad \rho_3 = \|x - a_3\|, \qquad \rho_4 = \|x - a_4\|.$$

Express these distance conditions as a set of three linear equations in the vector x. *Hint.* Square the distance equations, and subtract one from the others.

Solution. We square and expand the right-hand sides:

$$x^T x - 2a_i^T x + a_i^T a_i = \rho_i^2, \quad i = 1, 2, 3, 4.$$

The quadratic term $x^T x$ can be eliminated by subtracting one equation from the three others. For example,

$$\begin{bmatrix} -2(a_1 - a_4)^T \\ -2(a_2 - a_4)^T \\ -2(a_3 - a_4)^T \end{bmatrix} x = \begin{bmatrix} \rho_1^2 - \rho_4^2 - \|a_1\|^2 + \|a_4\|^2 \\ \rho_2^2 - \rho_4^2 - \|a_2\|^2 + \|a_4\|^2 \\ \rho_3^2 - \rho_4^2 - \|a_3\|^2 + \|a_4\|^2 \end{bmatrix}.$$

8.12 *Quadrature.* Consider a function $f : \mathbf{R} \to \mathbf{R}$. We are interested in *estimating* the definite integral $\alpha = \int_{-1}^{1} f(x)\, dx$ based on the value of f at some points $t_1, \ldots, t_n$. (We typically have $-1 \leq t_1 < t_2 < \cdots < t_n \leq 1$, but this is not needed here.) The standard method for estimating α is to form a weighted sum of the values $f(t_i)$:

$$\hat{\alpha} = w_1 f(t_1) + \cdots + w_n f(t_n),$$

where $\hat{\alpha}$ is our estimate of α, and $w_1, \ldots, w_n$ are the weights. This method of estimating the value of an integral of a function from its values at some points is a classical method in applied mathematics called *quadrature*. There are many quadrature methods (*i.e.*, choices of the points t_i and weights w_i). The most famous one is due to the mathematician Carl Friedrich Gauss, and bears his name.

(a) A typical requirement in quadrature is that the approximation should be exact (*i.e.*, $\hat{\alpha} = \alpha$) when f is any polynomial up to degree d, where d is given. In this case we say that the quadrature method has *order* d. Express this condition as a set of linear equations on the weights, $Aw = b$, assuming the points $t_1, \ldots, t_n$ are given. *Hint.* If $\hat{\alpha} = \alpha$ holds for the specific cases $f(x) = 1$, $f(x) = x$, $\ldots$, $f(x) = x^d$, then it holds for any polynomial of degree up to d.

(b) Show that the following quadrature methods have order 1, 2, and 3 respectively.

- *Trapezoid rule:* $n = 2$, $t_1 = -1$, $t_2 = 1$, and
$$w_1 = 1/2, \qquad w_2 = 1/2.$$
- *Simpson's rule:* $n = 3$, $t_1 = -1$, $t_2 = 0$, $t_3 = 1$, and
$$w_1 = 1/3, \qquad w_2 = 4/3, \qquad w_3 = 1/3.$$
(Named after the mathematician Thomas Simpson.)
- *Simpson's 3/8 rule:* $n = 4$, $t_1 = -1$, $t_2 = -1/3$, $t_3 = 1/3$, $t_4 = 1$,
$$w_1 = 1/4, \qquad w_2 = 3/4, \qquad w_3 = 3/4, \qquad w_4 = 1/4.$$

Solution.

(a) Let b_k be the exact integral of x^{k-1} for $k \geq 1$:

$$b_k = \int_{-1}^{1} t^{k-1} dt = \begin{cases} 2/k & k \text{ odd} \\ 0 & k \text{ even.} \end{cases}$$

Then the quadrature rule has order d if

$$w_1 t_1^{k-1} + w_2 t_2^{k-1} + \cdots + w_n t_n^{k-1} = b_k, \quad k = 1, 2, \ldots, d+1.$$

In matrix form,

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ t_1 & t_2 & \cdots & t_n \\ t_1^2 & t_2^2 & \cdots & t_n^2 \\ \vdots & \vdots & & \vdots \\ t_1^d & t_2^d & \cdots & t_n^d \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_{d+1} \end{bmatrix}.$$

(b) We verify that the weights satisfy the equation in part (a).

- *Trapezoid rule.*
$$\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 1/2 \\ 1/2 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \end{bmatrix}.$$
- *Simpson's rule.*
$$\begin{bmatrix} 1 & 1 & 1 \\ -1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1/3 \\ 4/3 \\ 1/3 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ 2/3 \end{bmatrix}.$$
- *Simpson's* 3/8 *rule.*
$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & -1/3 & 1/3 & 1 \\ 1 & 1/9 & 1/9 & 1 \\ -1 & -1/27 & 1/27 & 1 \end{bmatrix} \begin{bmatrix} 1/4 \\ 3/4 \\ 3/4 \\ 1/4 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ 2/3 \\ 0 \end{bmatrix}.$$

8.13 *Portfolio sector exposures.* (See exercise 1.14.) The n-vector h denotes a portfolio of investments in n assets, with h_i the dollar value invested in asset i. We consider a set of m industry sectors, such as pharmaceuticals or consumer electronics. Each asset is assigned to one of these sectors. (More complex models allow for an asset to be assigned to more than one sector.) The *exposure* of the portfolio to sector i is defined as the sum of investments in the assets in that sector. We denote the sector exposures using the m-vector s, where s_i is the portfolio exposure to sector i. (When $s_i = 0$, the portfolio is said to be *neutral* to sector i.) An investment advisor specifies a set of desired sector exposures, given as the m-vector s^{des}. Express the requirement $s = s^{\mathrm{des}}$ as a set of linear equations of the form $Ah = b$. (You must describe the matrix A and the vector b.)

Remark. A typical practical case involves $n = 1000$ assets and $m = 50$ sectors. An advisor might specify $s_i^{\mathrm{des}} = 0$ if she does not have an opinion as how companies in that sector will do in the future; she might specify a positive value for s_i^{des} if she thinks the companies in that sector will do well (*i.e.*, generate positive returns) in the future, and a negative value if she thinks they will do poorly.

Solution. The matrix A has size $m \times n$ and entries

$$A_{ij} = \begin{cases} 1 & \text{asset } j \text{ is assigned to sector } i \\ 0 & \text{otherwise.} \end{cases}$$

With this definition the ith entry of Ah is the sum of the investments h_j in assets j in sector i. The vector b is equal to s^{des}.

8.14 *Affine combinations of solutions of linear equations.* Consider the set of m linear equations in n variables $Ax = b$, where A is an $m \times n$ matrix, b is an m-vector, and x is the n-vector of variables. Suppose that the n-vectors $z_1, \ldots, z_k$ are solutions of this set of equations, *i.e.*, satisfy $Az_i = b$. Show that if the coefficients $\alpha_1, \ldots, \alpha_k$ satisfy $\alpha_1 + \cdots + \alpha_k = 1$, then the affine combination

$$w = \alpha_1 z_1 + \cdots + \alpha_k z_k$$

is a solution of the linear equations, *i.e.*, satisfies $Aw = b$. In words: Any affine combination of solutions of a set of linear equations is also a solution of the equations.

Solution.

$$\begin{aligned} Aw &= A(\alpha_1 z_1 + \cdots + \alpha_k z_k) \\ &= A(\alpha_1 z_1) + \cdots + A(\alpha_k z_k) \\ &= \alpha_1 A z_1 + \cdots + \alpha_k A z_k \\ &= \alpha_1 b + \cdots + \alpha_k b \\ &= (\alpha_1 + \cdots + \alpha_k) b \\ &= b. \end{aligned}$$

8.15 *Stoichiometry and equilibrium reaction rates.* We consider a system (such as a single cell) containing m metabolites (chemical species), with n reactions among the metabolites occurring at rates given by the n-vector r. (A negative reaction rate means the reaction runs in reverse.) Each reaction consumes some metabolites and produces others, in known rates proportional to the reaction rate. This is specified in the $m \times n$ *stoichiometry matrix* S, where S_{ij} is the rate of metabolite i production by reaction j, running at rate one. (When S_{ij} is negative, it means that when reaction j runs at rate one, metabolite i is consumed.) The system is said to be in equilibrium if the total production rate of each metabolite, due to all the reactions, is zero. This means that for each metabolite, the total production rate balances the total consumption rate, so the total quantities of the metabolites in the system do not change. Express the condition that the system is in equilibrium as a set of linear equations in the reaction rates.

Solution. The m-vector Sr gives the vector of metabolite production rates. So the system is in equilibrium when $Sr = 0$. This is a set of m equations with n variables.

8.16 *Bi-linear interpolation.* We are given a scalar value at each of the four corners of a square in 2-D, (x_1, y_1), (x_1, y_2), (x_2, y_1), and (x_2, y_2), where $x_1 < x_2$ and $y_1 < y_2$. We refer to these four values as F_{11}, F_{12}, F_{21}, and F_{22}, respectively. A *bi-linear interpolation* is a function of the form

$$f(u, v) = \theta_1 + \theta_2 u + \theta_3 v + \theta_4 uv,$$

where $\theta_1, \ldots, \theta_4$ are coefficients, that satisfies

$$f(x_1, y_1) = F_{11}, \quad f(x_1, y_2) = F_{12}, \quad f(x_2, y_1) = F_{21}, \quad f(x_2, y_2) = F_{22},$$

i.e., it agrees with (or interpolates) the given values on the four corners of the square. (The function f is usually evaluated only for points (u, v) inside the square. It is called bi-linear since it is affine in u when v is fixed, and affine in v when u is fixed.)

Express the interpolation conditions as a set of linear equations of the form $A\theta = b$, where A is a 4×4 matrix and b is a 4-vector. Give the entries of A and b in terms of x_1, x_2, y_1, y_2, F_{11}, F_{12}, F_{21}, and F_{22}.

Remark. Bi-linear interpolation is used in many applications to guess or approximate the values of a function at an arbitrary point in 2-D, given the function values on a grid of points. To approximate the value at a point (x, y), we first find the square of grid points that the point lies in. Then we use bi-linear interpolation to get the approximate value at (x, y).

Solution.

$$\begin{bmatrix} 1 & x_1 & y_1 & x_1 y_1 \\ 1 & x_1 & y_2 & x_1 y_2 \\ 1 & x_2 & y_1 & x_2 y_1 \\ 1 & x_2 & y_2 & x_2 y_2 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \end{bmatrix} = \begin{bmatrix} F_{11} \\ F_{12} \\ F_{21} \\ F_{22} \end{bmatrix}.$$

Chapter 9

Linear dynamical systems

Exercises

9.1 *Compartmental system.* A *compartmental system* is a model used to describe the movement of some material over time among a set of n compartments of a system, and the outside world. It is widely used in *pharmaco-kinetics*, the study of how the concentration of a drug varies over time in the body. In this application, the material is a drug, and the compartments are the bloodstream, lungs, heart, liver, kidneys, and so on. Compartmental systems are special cases of linear dynamical systems.

In this problem we will consider a very simple compartmental system with 3 compartments. We let $(x_t)_i$ denote the amount of the material (say, a drug) in compartment i at time period t. Between period t and period $t+1$, the material moves as follows.

- 10% of the material in compartment 1 moves to compartment 2. (This decreases the amount in compartment 1 and increases the amount in compartment 2.)
- 5% of the material in compartment 2 moves to compartment 3.
- 5% of the material in compartment 3 moves to compartment 1.
- 5% of the material in compartment 3 is eliminated.

Express this compartmental system as a linear dynamical system, $x_{t+1} = Ax_t$. (Give the matrix A.) Be sure to account for all the material entering and leaving each compartment.

Solution.

$$\begin{bmatrix} (x_{t+1})_1 \\ (x_{t+1})_2 \\ (x_{t+1})_3 \end{bmatrix} = \begin{bmatrix} 0.90 & 0 & 0.05 \\ 0.10 & 0.95 & 0 \\ 0 & 0.05 & 0.90 \end{bmatrix} \begin{bmatrix} (x_t)_1 \\ (x_t)_2 \\ (x_t)_3 \end{bmatrix}.$$

9.2 *Dynamics of an economy.* An economy (of a country or region) is described by an n-vector a_t, where $(a_t)_i$ is the economic output in sector i in year t (measured in billions of dollars, say). The total output of the economy in year t is $\mathbf{1}^T a_t$. A very simple model of how the economic output changes over time is $a_{t+1} = Ba_t$, where B is an $n \times n$ matrix. (This is closely related to the Leontief input-output model described on page 157 of the book. But the Leontief model is static, *i.e.*, doesn't consider how an economy changes over time.) The entries of a_t and B are positive in general.

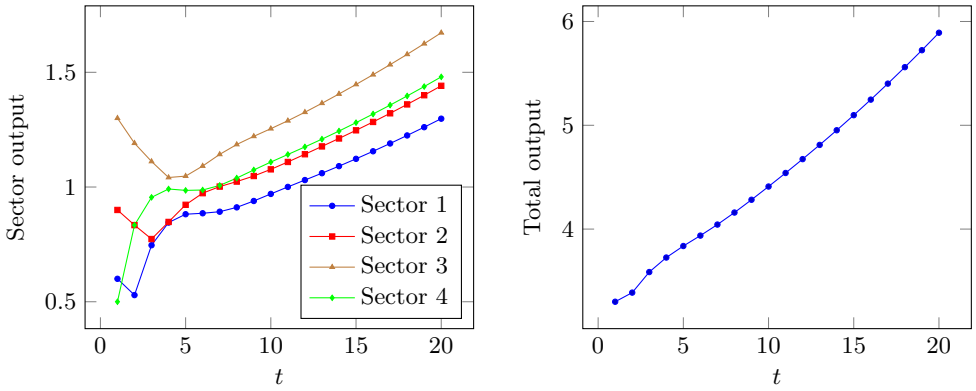
In this problem we will consider the specific model with $n = 4$ sectors and

$$B = \begin{bmatrix} 0.10 & 0.06 & 0.05 & 0.70 \\ 0.48 & 0.44 & 0.10 & 0.04 \\ 0.00 & 0.55 & 0.52 & 0.04 \\ 0.04 & 0.01 & 0.42 & 0.51 \end{bmatrix}.$$

(a) Briefly interpret B_{23}, in English.

(b) *Simulation.* Suppose $a_1 = (0.6, 0.9, 1.3, 0.5)$. Plot the four sector outputs (*i.e.*, $(a_t)_i$ for $i = 1, \ldots, 4$) and the total economic output (*i.e.*, $\mathbf{1}^T a_t$) versus t, for $t = 1, \ldots, 20$.

Solution. We interpret B_{23} as the factor by which economic output in sector 3 this year stimulates economic output in sector 2 next year. We might also say: Each dollar of economic output in sector 3 this year leads to B_{23} dollars of economic output next year.



9.3 *Equilibrium point for linear dynamical system.* Consider a time-invariant linear dynamical system with offset, $x_{t+1} = Ax_t + c$, where x_t is the state n-vector. We say that a vector z is an *equilibrium point* of the linear dynamical system if $x_1 = z$ implies $x_2 = z$, $x_3 = z$, ... (In words: If the system starts in state z, it stays in state z.)

Find a matrix F and vector g for which the set of linear equations $Fz = g$ characterizes equilibrium points. (This means: If z is an equilibrium point, then $Fz = g$; conversely if $Fz = g$, then z is an equilibrium point.) Express F and g in terms of A, c, any standard matrices or vectors (*e.g.*, I, $\mathbf{1}$, or 0), and matrix and vector operations.

Remark. Equilibrium points often have interesting interpretations. For example, if the linear dynamical system describes the population dynamics of a country, with the vector c denoting immigration (emigration when entries of c are negative), an equilibrium point is a population distribution that does not change, year to year. In other words, immigration exactly cancels the changes in population distribution caused by aging, births, and deaths.

Solution. The point z is an equilibrium point if and only if $z = Az + c$. We can rewrite this set of equations as

$$(I - A)z = c,$$

which has the required form with $F = I - A$, $g = c$. (There are several other answers, such as $F = A - I$, $g = -c$.)

9.4 *Reducing a Markov model to a linear dynamical system.* Consider the 2-Markov model

$$x_{t+1} = A_1 x_t + A_2 x_{t-1}, \quad t = 2, 3, \ldots,$$

where x_t is an n-vector. Define $z_t = (x_t, x_{t-1})$. Show that z_t satisfies the linear dynamical system equation $z_{t+1} = Bz_t$, for $t = 2, 3, \ldots$, where B is a $(2n) \times (2n)$ matrix. This idea can be used to express any K-Markov model as a linear dynamical system, with state $(x_t, \ldots, x_{t-K+1})$.

Solution.

$$z_{t+1} = \begin{bmatrix} A_1 & A_2 \\ I & 0 \end{bmatrix} z_t.$$

9.5 *Fibonacci sequence.* The Fibonacci sequence $y_0, y_1, y_2, \ldots$ starts with $y_0 = 0$, $y_1 = 1$, and for $t = 2, 3, \ldots$, y_t is the sum of the previous two entries, *i.e.*, $y_{t-1} + y_{t-2}$. (Fibonacci is the name used by the 13th century mathematician Leonardo of Pisa.) Express this as a time-invariant linear dynamical system with state $x_t = (y_t, y_{t-1})$ and output y_t, for $t = 1, 2, \ldots$. Use your linear dynamical system to simulate (compute) the Fibonacci sequence up to $t = 20$. Also simulate a modified Fibonacci sequence $z_0, z_1, z_2, \ldots$, which starts with the same values $z_0 = 0$ and $z_1 = 1$, but for $t = 2, 3, \ldots$, z_t is the difference of the two previous values, *i.e.*, $z_{t-1} - z_{t-2}$.

Solution. The Fibonacci sequence is described the linear dynamical system

$$x_{t+1} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} x_t, \qquad x_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

The values of y_t for $t = 0, \ldots, 20$ are

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181.

The modified Fibonacci sequence is described the linear dynamical system

$$x_{t+1} = \begin{bmatrix} 1 & -1 \\ 1 & 0 \end{bmatrix} z_t, \qquad x_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

The values of z_t for $t = 0, \ldots, 20$ are

$$0,\ 1,\ 1,\ 0,\ -1,\ -1,\ 0,\ 1,\ 1,\ 0,\ -1,\ -1,\ 0,\ 1,\ 1,\ 0,\ -1,\ -1,\ 0,\ 1.$$

9.6 *Recursive averaging.* Suppose that $u_1, u_2, \ldots$ is a sequence of n-vectors. Let $x_1 = 0$, and for $t = 2, 3, \ldots$, let x_t be the average of $u_1, \ldots, u_{t-1}$, *i.e.*, $x_t = (u_1 + \cdots + u_{t-1})/(t-1)$. Express this as a linear dynamical system with input, *i.e.*, $x_{t+1} = A_t x_t + B_t u_t$, $t = 1, 2, \ldots$ (with initial state $x_1 = 0$). *Remark.* This can be used to compute the average of an extremely large collection of vectors, by accessing them one-by-one.

Solution. The averages x_{t+1} and x_t are related as

$$x_{t+1} = \frac{1}{t}(u_1 + \cdots + u_t) = \frac{1}{t}((t-1)x_t + u_t) = \frac{t-1}{t}x_t + \frac{1}{t}u_t.$$

Hence

$$A_t = \frac{t-1}{t}I, \qquad B_t = \frac{1}{t}I.$$

9.7 *Complexity of linear dynamical system simulation.* Consider the time-invariant linear dynamical system with n-vector state x_t and m-vector input u_t, and dynamics $x_{t+1} = Ax_t + Bu_t$, $t = 1, 2, \ldots$. You are given the matrices A and B, the initial state x_1, and the inputs $u_1, \ldots, u_{T-1}$. What is the complexity of carrying out a simulation, *i.e.*, computing $x_2, x_3, \ldots, x_T$? About how long would it take to carry out a simulation with $n = 15$, $m = 5$, and $T = 10^5$, using a 1 Gflop/s computer?

Solution. The computation of x_{t-1} from x_t and u_t requires two matrix-vector products and one vector addition. The product Ax_t takes $(2n-1)n$ flops, the product Bu_t takes $(2m-1)n$ flops, and adding the two vectors takes n flops. The entire simulation takes

$$(T-1)((2n-1)n + (2m-1)n + n)$$

or approximately $2n(n+m)T$ flops.

For $n = 15$, $m = 5$, $T = 10^5$ this equals 6×10^7 flops. On a 1 Gflop/s computer the simulation would take about 6 milliseconds.

Chapter 10

Matrix multiplication

Exercises

10.1 *Scalar-row-vector multiplication.* Suppose a is a number and $x = [x_1 \ \cdots \ x_n]$ is an n-row-vector. The scalar-row-vector product ax is the n-row-vector $[ax_1 \ \cdots \ ax_n]$. Is this a special case of matrix-matrix multiplication? That is, can you interpret scalar-row-vector multiplication as matrix multiplication? (Recall that scalar-vector multiplication, with the scalar on the left, is *not* a special case of matrix-matrix multiplication; see page 177.)

Solution. Yes, scalar-row-vector multiplication is the same as matrix-matrix multiplication. We consider a a 1×1 matrix and x a $1 \times n$ matrix. The matrix-matrix product is

$$ax = \begin{bmatrix} a \end{bmatrix} \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix} = \begin{bmatrix} ax_1 & \cdots & ax_n \end{bmatrix}$$

and this is the same as the scalar-row-vector product.

10.2 *Ones matrix.* There is no special notation for an $m \times n$ matrix all of whose entries are one. Give a simple expression for this matrix in terms of matrix multiplication, transpose, and the ones vectors $\mathbf{1}_m$, $\mathbf{1}_n$ (where the subscripts denote the dimension).

Solution. The matrix is given by $\mathbf{1}_m \mathbf{1}_n^T$. This matrix is a product of an $m \times 1$ matrix and a $1 \times n$ matrix, so it has dimension $m \times n$. Its i, j entry is $1 \cdot 1 = 1$.

10.3 *Matrix sizes.* Suppose A, B, and C are matrices that satisfy $A + BB^T = C$. Determine which of the following statements are necessarily true. (There may be more than one true statement.)

(a) A is square.

(b) A and B have the same dimensions.

(c) A, B, and C have the same number of rows.

(d) B is a tall matrix.

Solution.

(a) True.

(b) Not necessarily true.

(c) True.

(d) Not necessarily true.

10.4 *Block matrix notation.* Consider the block matrix

$$A = \begin{bmatrix} I & B & 0 \\ B^T & 0 & 0 \\ 0 & 0 & BB^T \end{bmatrix},$$

where B is 10×5. What are the dimensions of the four zero matrices and the identity matrix in the definition of A? What are the dimensions of A?

Solution.

$$A = \begin{bmatrix} I_{10} & B & 0_{10\times 10} \\ B^T & 0_{5\times 5} & 0_{5\times 10} \\ 0_{10\times 10} & 0_{10\times 5} & BB^T \end{bmatrix}.$$

A is 25×25.

Here's one way to work out these dimensions. They all depend on the requirement that submatrices in a block row have the same height (number of rows) and submatrices in a block column have the same width (number of columns).

First, the height of I and the height of B must the same, so the height of I is 10. It's a square matrix, so the I matrix is 10×10. The matrix BB^T is the 3,3 position is also 10×10, and it must have the same number of columns as the two zero matrices above it. So they both have 10 columns. The zero matrices in the 3rd block row must have the

same number of rows as BB^T, *i.e.*, 10. The height of the zero matrix in the 1,3 position is the same as the height of B, *i.e.*, 10. So the 1,3 zero matrix is 10×10. The height of the zero matrix in the 2,3 position is the same as B^T, *i.e.*, 5. So it is 5×10. Similar arguments can be used to deduce the dimensions of the other zero matrices.

10.5 *When is the outer product symmetric?* Let a and b be n-vectors. The inner product is symmetric, *i.e.*, we have $a^T b = b^T a$. The outer product of the two vectors is generally *not* symmetric; that is, we generally have $ab^T \neq ba^T$. What are the conditions on a and b under which $ab = ba^T$? You can assume that all the entries of a and b are nonzero. (The conclusion you come to will hold even when some entries of a or b are zero.) *Hint.* Show that $ab^T = ba^T$ implies that a_i/b_i is a constant (*i.e.*, independent of i).

Solution. The condition $ab^T = ba^T$ can be expressed as

$$a_i b_j = b_i a_j, \quad i, j = 1, \ldots, n.$$

Assuming that all entries of a and b are nonzero, we divide the equation above by $a_j b_i$ to find that $a_i/b_i = a_j/b_j$ for all i, j. So it is a constant, let's say α. Then we see that $a = \alpha b$, *i.e.*, one of the vectors is a multiple of the other.

10.6 *Product of rotation matrices.* Let A be the 2×2 matrix that corresponds to rotation by θ radians, defined in (7.1), and let B be the 2×2 matrix that corresponds to rotation by ω radians. Show that AB is also a rotation matrix, and give the angle by which it rotates vectors. Verify that $AB = BA$ in this case, and give a simple English explanation.

Solution. The product AB is

$$\begin{aligned} AB &= \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\omega & -\sin\omega \\ \sin\omega & \cos\omega \end{bmatrix} \\ &= \begin{bmatrix} \cos\theta\cos\omega - \sin\theta\sin\omega & -\cos\theta\sin\omega - \sin\theta\cos\omega \\ \sin\theta\cos\omega + \sin\theta\cos\omega & -\sin\theta\sin\omega + \cos\theta\cos\omega \end{bmatrix} \\ &= \begin{bmatrix} \cos(\theta+\omega) & -\sin(\theta+\omega) \\ \sin(\theta+\omega) & \cos(\theta+\omega) \end{bmatrix}. \end{aligned}$$

AB is rotation by $\theta + \omega$ radians, which makes sense. For any vector x, Bx is x rotated ω radians; and $A(Bx) = (AB)x$ is the result, rotated by an additional θ radians.

We verify that $BA = AB$:

$$\begin{aligned} BA &= \begin{bmatrix} \cos\omega & -\sin\omega \\ \sin\omega & \cos\omega \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \\ &= \begin{bmatrix} \cos\omega\cos\theta - \sin\omega\sin\theta & -\cos\omega\sin\theta - \sin\omega\cos\theta \\ \sin\omega\cos\theta + \sin\omega\cos\theta & -\sin\omega\sin\theta + \cos\omega\cos\theta \end{bmatrix} \\ &= \begin{bmatrix} \cos(\omega+\theta) & -\sin(\omega+\theta) \\ \sin(\omega+\theta) & \cos(\omega+\theta) \end{bmatrix}. \end{aligned}$$

We can explain $AB = BA$ by saying that it does not matter in which order you rotate by θ and by ω radians; the result is the same.

10.7 *Two rotations.* Two 3-vectors x and y are related as follows. First, the vector x is rotated 40° around the e_3 axis, counterclockwise (from e_1 toward e_2), to obtain the 3-vector z. Then, z is rotated 20° around the e_1 axis, counterclockwise (from e_2 toward e_3), to form y. Find the 3×3 matrix A for which $y = Ax$. Verify that A is an orthogonal matrix. *Hint.* Express A as a product of two matrices, which carry out the two rotations described above.

Solution. The matrix that rotates x into z (*i.e.*, $z = Ux$) is given by

$$U = \begin{bmatrix} \cos 40^\circ & -\sin 40^\circ & 0 \\ \sin 40^\circ & \cos 40^\circ & 0 \\ 0 & & 1 \end{bmatrix} = \begin{bmatrix} 0.766 & -0.643 & 0 \\ 0.643 & 0.766 & 0 \\ 0 & & 1 \end{bmatrix}.$$

The matrix that rotates z into y (*i.e.*, $y = Vx$) is given by

$$V = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos 20^\circ & -\sin 20^\circ \\ 0 & \sin 20^\circ & \cos 20^\circ \end{bmatrix} = \begin{bmatrix} 1.000 & 0 & 0 \\ 0 & 0.940 & -0.342 \\ 0 & 0.342 & 0.940 \end{bmatrix}.$$

We have $y = VUx$, so

$$A = VU = \begin{bmatrix} 0.766 & -0.643 & 0.000 \\ 0.604 & 0.720 & -0.342 \\ 0.220 & 0.262 & 0.940 \end{bmatrix}.$$

We compute A^TA, and find that it is I (or very close; there are small errors due to floating-point rounding). The matrices U and V are also orthogonal.

10.8 *Entries of matrix triple product.* (See page 182.) Suppose A has dimensions $m \times n$, B has dimensions $n \times p$, C has dimensions $p \times q$, and let $D = ABC$. Show that

$$D_{ij} = \sum_{k=1}^{n} \sum_{l=1}^{p} A_{ik} B_{kl} C_{lj}.$$

This is the formula analogous to (10.1) for the product of two matrices.

Solution.

$$\begin{aligned} D_{ij} &= \sum_{k=1}^{n} A_{ik} (BC)_{kj} \\ &= \sum_{k=1}^{n} A_{ik} \left(\sum_{l=1}^{p} B_{kl} C_{lj} \right) \\ &= \sum_{k=1}^{n} \sum_{l=1}^{p} A_{ik} B_{kl} C_{lj}. \end{aligned}$$

10.9 *Multiplication by a diagonal matrix.* Suppose that A is an $m \times n$ matrix, D is a diagonal matrix, and $B = DA$. Describe B in terms of A and the entries of D. You can refer to the rows or columns or entries of A.

Solution. The ith row of B is the ith row of A, scaled by D_{ii}.

To see, consider the i, j entry of $B = DA$:

$$B_{ij} = \sum_{k=1}^{m} D_{ik} A_{kj} = D_{ii} A_{ij}.$$

The first equality is the definition of a general matrix-matrix product. The second equality follows because $D_{ik} = 0$ for $k \neq i$.

10.10 *Converting from purchase quantity matrix to purchase dollar matrix.* An $n \times N$ matrix Q gives the purchase history of a set of n products by N customers, over some period, with Q_{ij} being the quantity of product i bought by customer j. The n-vector p gives the product prices. A data analyst needs the $n \times N$ matrix D, where D_{ij} is the total dollar value that customer j spent on product i. Express D in terms of Q and p, using compact matrix/vector notation. You can use any notation or ideas we have encountered, *e.g.*, stacking, slicing, block matrices, transpose, matrix-vector product, matrix-matrix product, inner product, norm, correlation, **diag**(), and so on.

Solution. We have $D = \mathbf{diag}(p)Q$. To see how we get this, we note that the dollar spending of customer j on product i is the quantity they bought, Q_{ij}, times the price of the product, p_i, so $D_{ij} = p_i Q_{ij}$. This is expressed in matrix notation as $D = \mathbf{diag}(p)Q$.

10.11 *Trace of matrix-matrix product.* The sum of the diagonal entries of a square matrix is called the *trace* of the matrix, denoted $\mathbf{tr}(A)$.

(a) Suppose A and B are $m \times n$ matrices. Show that

$$\mathbf{tr}(A^T B) = \sum_{i=1}^{m} \sum_{j=1}^{n} A_{ij} B_{ij}.$$

What is the complexity of calculating $\mathbf{tr}(A^T B)$?

(b) The number $\mathbf{tr}(A^T B)$ is sometimes referred to as the inner product of the matrices A and B. (This allows us to extend concepts like angle to matrices.) Show that $\mathbf{tr}(A^T B) = \mathbf{tr}(B^T A)$.

(c) Show that $\mathbf{tr}(A^T A) = \|A\|^2$. In other words, the square of the norm of a matrix is the trace of its Gram matrix.

(d) Show that $\mathbf{tr}(A^T B) = \mathbf{tr}(B A^T)$, even though in general $A^T B$ and $B A^T$ can have different dimensions, and even when they have the same dimensions, they need not be equal.

Solution.

(a) The diagonal entries of $A^T B$ are

$$(A^T B)_{jj} = \sum_{i=1}^{m} (A^T)_{ji} B_{ij} = \sum_{i=1}^{m} A_{ij} B_{ij}.$$

The complexity is $2mn$ flops. We need mn multiplications and $mn - 1$ additions. Note that this is lower than the $2mn^2$ complexity of the entire matrix-matrix product $A^T B$.

(b) The matrix $B^T A$ is the transpose of $A^T B$, so it has the same diagonal entries and trace.

(c) From part (a), $A^T A = \sum_{i=1}^{m} \sum_{j=1}^{n} A_{ij}^2 = \|A\|^2$.

(d) We have

$$\mathbf{tr}(B A^T) = \sum_{i=1}^{m} (B A^T)_{ii} = \sum_{i=1}^{m} \sum_{j=1}^{n} B_{ij} (A^T)_{ji} = \sum_{i=1}^{m} \sum_{j=1}^{n} B_{ij} A_{ij},$$

the same expression as in part (a).

10.12 *Norm of matrix product.* Suppose A is an $m \times p$ matrix and B is a $p \times n$ matrix. Show that $\|AB\| \leq \|A\| \|B\|$, *i.e.*, the (matrix) norm of the matrix product is no more than the product of the norms of the matrices. *Hint.* Let $a_1^T, \ldots, a_m^T$ be the rows of A, and $b_1, \ldots, b_n$ be the columns of B. Then

$$\|AB\|^2 = \sum_{i=1}^{m} \sum_{j=1}^{n} (a_i^T b_j)^2.$$

Now use the Cauchy–Schwarz inequality.

Solution. The Cauchy–Schwarz inequality tells us that $|a_i^T b_j| \leq \|a_i\| \|b_j\|$, so $(a_i^T b_j)^2 \leq \|a_i\|^2 \|b_j\|^2$. Using the hint it follows that

$$\|AB\|^2 \quad = \quad \sum_{i=1}^{m} \sum_{j=1}^{n} (a_i^T b_j)^2$$

$$\begin{aligned} &\leq \sum_{i=1}^{m}\sum_{j=1}^{n}\|a_i\|^2\|b_j\|^2 \\ &= \left(\sum_{i=1}^{m}\|a_i\|^2\right)\left(\sum_{j=1}^{n}\|b_j\|^2\right) \\ &= \|A\|^2\|B\|^2. \end{aligned}$$

Taking the squareroot we get $\|AB\| \leq \|A\|\|B\|$.

10.13 *Laplacian matrix of a graph.* Let A be the incidence matrix of a directed graph with n nodes and m edges (see §7.3). The *Laplacian matrix* associated with the graph is defined as $L = AA^T$, which is the Gram matrix of A^T. It is named after the mathematician Pierre-Simon Laplace.

(a) Show that $\mathcal{D}(v) = v^T L v$, where $\mathcal{D}(v)$ is the Dirichlet energy defined on page 135.

(b) Describe the entries of L. *Hint.* The following two quantities might be useful: The *degree* of a node, which is the number of edges that connect to the node (in either direction), and the number of edges that connect a pair of distinct nodes (in either direction).

Solution.

(a) We have

$$v^T L v = v^T A A^T v = (A^T v)^T (A^T v) = \|A^T v\|^2 = \mathcal{L}(v).$$

(b) L_{ii} is the degree of node i. L_{ij} is minus the total number of edges that connect nodes i and j, in either direction. (Note that the direction of the edge connecting i and j does not affect L_{ij}.)

To see this, we first consider the expression for L_{ii}

$$L_{ii} = (AA^T)_{ii} = \sum_{k=1}^{m} A_{ik}^2.$$

Each term A_{ik}^2 in the sum is zero or one. It is one if edge k arrives at node i ($A_{ik} = 1$) or leaves node i ($A_{ik} = -1$), and it is zero if edge k is not incident to node i. Therefore L_{ii} is the degree of node i (the total number of edges connected to node i).

Suppose $i \neq j$. Then

$$L_{ij} = (AA^T)_{ij} = \sum_{k=1}^{m} A_{ik}A_{jk}.$$

Each term is a product of two entries in column k of the incidence matrix. Now, each column has exactly two nonzero entries, $A_{ik} = 1$ if i is the node at which edge j ends and $A_{ik} = -1$ if i is the node at which it leaves. The kth term $A_{ik}A_{jk}$ in the sum is therefore -1 or 0. It is equal to -1 only if edge k is between nodes i and j. By summing over k we obtain the negative of the number of edges between nodes i and j.

10.14 *Gram matrix.* Let $a_1, \ldots, a_n$ be the columns of the $m \times n$ matrix A. Suppose that the columns all have norm one, and for $i \neq j$, $\angle(a_i, a_j) = 60^\circ$. What can you say about the Gram matrix $G = A^T A$? Be as specific as you can be.

Solution. Since $\|a_j\|^2 = 1$, we conclude that $G_{jj} = 1$, *i.e.*, the diagonal entries of G are all one. For $i \neq j$, we have $\angle(a_i, a_j) = 60^\circ$, which means that $a_i^T a_j = 1/2$, and so $G_{ij} = 1/2$. So the Gram matrix satisfies

$$G_{ij} = \begin{cases} 1 & i = j, \\ 1/2 & i \neq j. \end{cases}$$

It can be written as $G = (1/2)I + (1/2)\mathbf{1}\mathbf{1}^T$.

10.15 *Pairwise distances from Gram matrix.* Let A be an $m \times n$ matrix with columns $a_1, \ldots, a_n$, and associated Gram matrix $G = A^T A$. Express $\|a_i - a_j\|$ in terms of G, specifically G_{ii}, G_{ij}, and G_{jj}.

Solution. We have $G_{ij} = a_i^T a_j$. Therefore

$$\|a_i - a_j\|^2 = a_i^T a_i - 2a_i^T a_j + a_j^T a_j = G_{ii} - 2G_{ij} + G_{jj}.$$

It follows that $\|a_i - a_j\| = (G_{ii} - 2G_{ij} + G_{jj})^{1/2}$.

10.16 *Covariance matrix.* Consider a list of k n-vectors $a_1, \ldots, a_k$, and define the $n \times k$ matrix $A = [a_1 \ \cdots \ a_k]$.

(a) Let the k-vector μ give the means of the columns, *i.e.*, $\mu_i = \mathbf{avg}(a_i)$, $i = 1, \ldots, k$. (The symbol μ is a traditional one to denote an average value.) Give an expression for μ in terms of the matrix A.

(b) Let $\tilde{a}_1, \ldots, \tilde{a}_k$ be the de-meaned versions of $a_1, \ldots, a_k$, and define $\tilde{A}$ as the $n \times k$ matrix $\tilde{A} = [\tilde{a}_1 \ \cdots \ \tilde{a}_k]$. Give a matrix expression for $\tilde{A}$ in terms of A and μ.

(c) The *covariance matrix* of the vectors $a_1, \ldots, a_k$ is the $k \times k$ matrix $\Sigma = (1/N)\tilde{A}^T\tilde{A}$, the Gram matrix of $\tilde{A}$ multiplied with $1/N$. Show that

$$\Sigma_{ij} = \begin{cases} \mathbf{std}(a_i)^2 & i = j \\ \mathbf{std}(a_i)\,\mathbf{std}(a_j)\rho_{ij} & i \neq j \end{cases}$$

where ρ_{ij} is the correlation coefficient of a_i and a_j. (The expression for $i \neq j$ assumes that ρ_{ij} is defined, *i.e.*, $\mathbf{std}(a_i)$ and $\mathbf{std}(a_j)$ are nonzero. If not, we interpret the formula as $\Sigma_{ij} = 0$.) Thus the covariance matrix encodes the standard deviations of the vectors, as well as correlations between all pairs. The correlation matrix is widely used in probability and statistics.

(d) Let $z_1, \ldots, z_k$ be the standardized versions of $a_1, \ldots, a_k$. (We assume the de-meaned vectors are nonzero.) Derive a matrix expression for $Z = [z_1 \ \cdots \ z_k]$, the matrix of standardized vectors. Your expression should use A, μ, and the numbers $\mathbf{std}(a_1), \ldots, \mathbf{std}(a_k)$.

Solution.

(a) $\mu = (1/n)A^T\mathbf{1}$.
This follows from $\mu_i = (\mathbf{1}^T a_i)/n = (a_i^T\mathbf{1})/n$.

(b) $\tilde{A} = A - \mathbf{1}\mu^T$.
The ith column of $\tilde{A}$ is $\tilde{a}_i = a_i - \mu_i\mathbf{1}$, so

$$\begin{aligned} \tilde{A} &= \begin{bmatrix} a_1 & \cdots & a_k \end{bmatrix} - \begin{bmatrix} \mu_1\mathbf{1} & \cdots & \mu_k\mathbf{1} \end{bmatrix} \\ &= \begin{bmatrix} a_1 & \cdots & a_k \end{bmatrix} - \mathbf{1}\begin{bmatrix} \mu_1 & \cdots & \mu_k \end{bmatrix} \\ &= A - \mathbf{1}\mu^T. \end{aligned}$$

(c) The diagonal entries of Σ_{ii} are

$$\Sigma_{ii} = \frac{1}{N}(\tilde{A}^T\tilde{A})_{ii} = \frac{1}{N}\tilde{a}_i\tilde{a}_i^T = \frac{1}{N}\|\tilde{a}_i\|^2 = \mathbf{std}(a_i)^2,$$

since the standard deviation is defined as $\mathbf{std}(a_i) = \|\tilde{a}_i\|/\sqrt{N}$.
The off-diagonal entry Σ_{ij} is zero if $\tilde{a}_i = 0$ or $\tilde{a}_j = 0$. Otherwise,

$$\Sigma_{ij} = \frac{1}{N}(\tilde{A}^T\tilde{A})_{ij} = \frac{1}{N}\tilde{a}_i\tilde{a}_j^T = \rho_{ij}\,\mathbf{std}(a_i)\,\mathbf{std}(a_j),$$

since the correlation coefficient is defined as

$$\rho_{ij} = \frac{1}{N} \frac{\tilde{a}_i^T \tilde{a}_j}{\mathbf{std}(a_i)\,\mathbf{std}(a_j)}.$$

(d) $Z = (A - \mathbf{1}\mu^T)\,\mathbf{diag}(1/\mathbf{std}(a_1), \ldots, 1/\mathbf{std}(a_k))$. The standardized vectors are $z_i = \tilde{a}_i/\mathbf{std}(a_i)$. Therefore

$$\begin{aligned} Z &= \tilde{A}\,\mathbf{diag}(1/\mathbf{std}(a_1), \ldots, 1/\mathbf{std}(a_k)) \\ &= (A - \mathbf{1}\mu^T)\,\mathbf{diag}(1/\mathbf{std}(a_1), \ldots, 1/\mathbf{std}(a_k)). \end{aligned}$$

10.17 *Patients and symptoms.* Each of a set of N patients can exhibit any number of a set of n symptoms. We express this as an $N \times n$ matrix S, with

$$S_{ij} = \begin{cases} 1 & \text{patient } i \text{ exhibits symptom } j \\ 0 & \text{patient } i \text{ does not exhibit symptom } j. \end{cases}$$

Give simple English descriptions of the following expressions. Include the dimensions, and describe the entries.

(a) $S\mathbf{1}$.

(b) $S^T\mathbf{1}$.

(c) S^TS.

(d) SS^T.

Solution.

(a) $S\mathbf{1}$ is an N-vector whose ith element is the total number of symptoms patient i has.

(b) $S^T\mathbf{1}$ is an n-vector whose ith element is the total number of patients who exhibit symptom i.

(c) S^TS is an $n \times n$ matrix whose entry in the ith row and jth column is the number of patients who exhibit both symptom i and symptom j.

(d) SS^T is an $N \times N$ matrix whose entry in the ith row and jth column is the number of symptoms patient i and patient j have in common.

10.18 *Students, classes, and majors.* We consider m students, n classes, and p majors. Each student can be in any number of the classes (although we'd expect the number to range from 3 to 6), and can have any number of the majors (although the common values would be 0, 1, or 2). The data about the students' classes and majors are given by an $m \times n$ matrix C and an $m \times p$ matrix M, where

$$C_{ij} = \begin{cases} 1 & \text{student } i \text{ is in class } j \\ 0 & \text{student } i \text{ is not in class } j, \end{cases}$$

and

$$M_{ij} = \begin{cases} 1 & \text{student } i \text{ is in major } j \\ 0 & \text{student } i \text{ is not in major } j. \end{cases}$$

(a) Let E be the n-vector with E_i being the enrollment in class i. Express E using matrix notation, in terms of the matrices C and M.

(b) Define the $n \times p$ matrix S where S_{ij} is the total number of students in class i with major j. Express S using matrix notation, in terms of the matrices C and M.

Solution.

(a) $E = C^T\mathbf{1}$. This follows from

$$E_i = \sum_{k=1}^{n} C_{ki} = \sum_{k=1}^{n} (C^T)_{ik} = (C^T\mathbf{1})_i.$$

(b) $S = C^T M$. This follows from

$$S_{ij} = \sum_{k=1}^{n} C_{ki}M_{kj} = \sum_{k=1}^{n} (C^T)_{ik}M_{kj} = (C^T M)_{ij}.$$

10.19 *Student group membership.* Let $G \in \mathbf{R}^{m\times n}$ represent a contingency matrix of m students who are members of n groups:

$$G_{ij} = \begin{cases} 1 & \text{student } i \text{ is in group } j \\ 0 & \text{student } i \text{ is not in group } j. \end{cases}$$

(A student can be in any number of the groups.)

(a) What is the meaning of the 3rd column of G?

(b) What is the meaning of the 15th row of G?

(c) Give a simple formula (using matrices, vectors, etc.) for the n-vector M, where M_i is the total membership (*i.e.*, number of students) in group i.

(d) Interpret $(GG^T)_{ij}$ in simple English.

(e) Interpret $(G^TG)_{ij}$ in simple English.

Solution.

(a) The roster for the 3rd group.

(b) The list of groups student 15 is in.

(c) $M = G^T\mathbf{1}$.

(d) $(GG^T)_{ij}$ is the number of groups student i and student j have in common. (Sanity check: $GG^T \in \mathbf{R}^{m\times m}$)

(e) $(G^TG)_{ij}$ is the number of students group i and group j have in common. (Sanity check: $G^TG \in \mathbf{R}^{n\times n}$)

10.20 *Products, materials, and locations.* P different products each require some amounts of M different materials, and are manufactured in L different locations, which have different material costs. We let C_{lm} denote the cost of material m in location l, for $l = 1, \ldots, L$ and $m = 1, \ldots, M$. We let Q_{mp} denote the amount of material m required to manufacture one unit of product p, for $m = 1, \ldots, M$ and $p = 1, \ldots, P$. Let T_{pl} denote the total cost to manufacture product p in location l, for $p = 1, \ldots, P$ and $l = 1, \ldots, L$. Give an expression for the matrix T.

Solution. $T = (CQ)^T = Q^TC^T$.

10.21 *Integral of product of polynomials.* Let p and q be two quadratic polynomials, given by

$$p(x) = c_1 + c_2x + c_3x^2, \qquad q(x) = d_1 + d_2x + d_3x^2.$$

Express the integral $J = \int_0^1 p(x)q(x)\,dx$ in the form $J = c^TGd$, where G is a 3×3 matrix. Give the entries of G (as numbers).

Solution. If we expand the product $p(x)q(x)$ we get

$$p(x)q(x) = c_1d_1 + (c_1d_2 + c_2d_1)x + (c_1d_3 + c_2d_2 + c_3d_1)x^2 + (c_2d_3 + c_3d_2)x^3 + (c_3d_3)x^4.$$

Integrating term by term gives

$$\begin{aligned} J &= \int_0^1 p(x)q(x)dx \\ &= c_1d_1 + \frac{1}{2}(c_1d_2 + c_2d_1) + \frac{1}{3}(c_1d_3 + c_2d_2 + c_3d_1) + \frac{1}{4}(c_2d_3 + c_3d_2) + \frac{1}{5}(c_3d_3). \end{aligned}$$

To write this in the form c^TGd we first write J as an inner product of c with Gd:

$$\begin{aligned} J &= c_1(d_1 + d_2/2 + d_3/3) + c_2(d_1/2 + d_2/3 + d_3/4) + c_3(d_1/3 + d_2/4 + d_3/5) \\ &= \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}^T \begin{bmatrix} d_1 + d_2/2 + d_3/3 \\ d_1/2 + d_2/3 + d_3/4 \\ d_1/3 + d_2/4 + d_3/5 \end{bmatrix}. \end{aligned}$$

The second vector is Gd with

$$G = \begin{bmatrix} 1 & 1/2 & 1/3 \\ 1/2 & 1/3 & 1/4 \\ 1/3 & 1/4 & 1/5 \end{bmatrix}.$$

An alternative method is to first write the product polynomial as

$$p(x)q(x) = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}^T \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}^T \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}^T \begin{bmatrix} 1 & x & x^2 \\ x & x^2 & x^3 \\ x^2 & x^3 & x^4 \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix}.$$

The integral is

$$\begin{aligned} \int_0^1 p(x)q(x)dx &= \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}^T \begin{bmatrix} \int_0^1 1dx & \int_0^1 xdx & \int_0^1 x^2dx \\ \int_0^1 xdx & \int_0^1 x^2dx & \int_0^1 x^3dx \\ \int_0^1 x^2dx & \int_0^1 x^3dx & \int_0^1 x^4dx \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} \\ &= \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}^T \begin{bmatrix} 1 & 1/2 & 1/3 \\ 1/2 & 1/3 & 1/4 \\ 1/3 & 1/4 & 1/5 \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix}. \end{aligned}$$

10.22 *Composition of linear dynamical systems.* We consider two time-invariant linear dynamical systems with outputs. The first one is given by

$$x_{t+1} = Ax_t + Bu_t, \qquad y_t = Cx_t, \quad t = 1, 2, \ldots,$$

with state x_t, input u_t, and output y_t. The second is given by

$$\tilde{x}_{t+1} = \tilde{A}\tilde{x}_t + \tilde{B}w_t, \qquad v_t = \tilde{C}\tilde{x}_t, \quad t = 1, 2, \ldots,$$

with state $\tilde{x}_t$, input w_t, and output v_t. We now connect the output of the first linear dynamical system to the input of the second one, which means we take $w_t = y_t$. (This is called the *composition* of the two systems.) Show that this composition can also be expressed as a linear dynamical system with state $z_t = (x_t, \tilde{x}_t)$, input u_t, and output v_t. (Give the state transition matrix, input matrix, and output matrix.)

Solution. Substituting Cx_t for w_t in the state transition equation for the second system gives

$$\tilde{x}_{t+1} = \tilde{A}\tilde{x}_t + \tilde{B}Cx_t.$$

The composition is described by

$$\begin{bmatrix} x_{t+1} \\ \tilde{x}_{t+1} \end{bmatrix} = \begin{bmatrix} A & 0 \\ \tilde{B}C & \tilde{A} \end{bmatrix} \begin{bmatrix} x_t \\ \tilde{x}_t \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u_t, \qquad v_t = \begin{bmatrix} 0 & \tilde{C} \end{bmatrix} \begin{bmatrix} x_t \\ \tilde{x}_t \end{bmatrix}.$$

This is a linear dynamical system with state $z_t = (x_t, \tilde{x}_t)$, and the following state transition matrix, input matrix, and output matrix:

$$\begin{bmatrix} A & 0 \\ \tilde{B}C & \tilde{A} \end{bmatrix}, \qquad \begin{bmatrix} B \\ 0 \end{bmatrix}, \qquad \begin{bmatrix} 0 & \tilde{C} \end{bmatrix}.$$

10.23 Suppose A is an $n \times n$ matrix that satisfies $A^2 = 0$. Does this imply that $A = 0$? (This is the case when $n = 1$.) If this is (always) true, explain why. If it is not, give a specific counterexample, *i.e.*, a matrix A that is nonzero but satisfies $A^2 = 0$.

Solution. No. The following matrix is a counterexample:

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

A is nonzero but satisfies $A^2 = 0$.

10.24 *Matrix power identity.* A student says that for any square matrix A,

$$(A + I)^3 = A^3 + 3A^2 + 3A + I.$$

Is she right? If she is, explain why; if she is wrong, give a specific counterexample, *i.e.*, a square matrix A for which it does not hold.

Solution. The student is right. Expanding the power, we have

$$\begin{aligned} (A + I)^3 &= (A + I)(A + I)(A + I) \\ &= (A^2 + IA + AI + I^2)(A + I) \\ &= (A^2 + 2A + I)(A + I) \\ &= A^3 + 3A^2 + 3A + I. \end{aligned}$$

This all works out because the matrices A and I in the expression commute, *i.e.*, satisfy $IA = AI$. For general A and B, it is false that $(A + B)^3 = A^3 + 3A^2B + 3AB^2 + B^3$.

10.25 *Squareroots of the identity.* The number 1 has two squareroots (*i.e.*, numbers who square is 1), 1 and -1. The $n \times n$ identity matrix I_n has many more squareroots.

(a) Find all diagonal squareroots of I_n. How many are there? (For $n = 1$, you should get 2.)

(b) Find a nondiagonal 2×2 matrix A that satisfies $A^2 = I$. This means that in general there are even more squareroots of I_n than you found in part (a).

Solution.

(a) The diagonal squareroots are the diagonal matrices with diagonal entries 1 or -1. The $n \times n$ identity matrix has 2^n diagonal squareroots.

The square of a diagonal matrix D is the diagonal matrix with D_{ii}^2 as ith diagonal entry. Therefore D is a diagonal squareroot of I_n if and only if $D_{ii}^2 = 1$ for $i = 1, \ldots, n$, *i.e.*, $D_{ii} = 1$ or $D_{ii} = -1$.

(b) $A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$.

10.26 *Circular shift matrices.* Let A be the 5×5 matrix

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

(a) How is Ax related to x? Your answer should be in English. *Hint.* See exercise title.

(b) What is A^5? *Hint.* The answer should make sense, given your answer to part (a).

Solution.

(a) Ax is a circular downward shift over one position:

$$Ax = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} x_5 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}.$$

The entries of x and shifted down on position, and the last entry comes around to the first entry.

(b) A^k is a circular shift over k positions.

$$A^2 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}, \qquad A^3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$$A^4 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad A^5 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

A^5 is the identity matrix. If we circularly shift x over five positions, we obtain x again.

10.27 *Dynamics of an economy.* Let $x_1, x_2, \ldots$ be n-vectors that give the level of economic activity of a country in years $1, 2, \ldots$, in n different sectors (like energy, defense, manufacturing). Specifically, $(x_t)_i$ is the level of economic activity in economic sector i (say, in billions of dollars) in year t. A common model that connects these economic activity vectors is $x_{t+1} = Bx_t$, where B is an $n \times n$ matrix. (See exercise 9.2.)

Give a matrix expression for the total economic activity across all sectors in year $t = 6$, in terms of the matrix B and the vector of initial activity levels x_1. Suppose you can increase economic activity in year $t = 1$ by some fixed amount (say, one billion dollars) in *one* sector, by government spending. How should you choose which sector to stimulate so as to maximize the total economic output in year $t = 6$?

Solution. We have $x_2 = Bx_1$, and continuing, we get $x_{11} = B^5 x_1$. This n-vector gives the economic activity level in all n sectors. To get the total, we sum them, by multiplying this vector on the left by $\mathbf{1}^T$. This gives the correct expression: $\mathbf{1}^T(B^5)x_1$.

We can write this as $c^T x_1$, where $c = (B^5)^T \mathbf{1}$. When we increase sector i output at $t = 1$ by the amount δ, the effect on the total economic output at $t = 6$ is δc_i. To get the largest increase for a fixed δ, we choose i for which c_i is largest.

10.28 *Controllability matrix.* Consider the time-invariant linear dynamical system $x_{t+1} = Ax_t + Bu_t$, with n-vector state x_t and m-vector input u_t. Let $U = (u_1, u_2, \ldots, u_{T-1})$ denote the sequence of inputs, stacked in one vector. Find the matrix C_T for which

$$x_T = A^{T-1}x_1 + C_T U$$

holds. The first term is what x_T would be if $u_1 = \cdots = u_{T-1} = 0$; the second term shows how the sequence of inputs $u_1, \ldots, u_{T-1}$ affect x_T. The matrix C_T is called the *controllability matrix* of the linear dynamical system.

Solution. We use $x_t = Ax_t + Bu_t$ recursively to find an expression for x_T:

$$\begin{aligned} x_2 &= Ax_1 + Bu_1 \\ x_3 &= Ax_2 + Bu_2 \\ &= A^2x_1 + ABu_1 + Bu_2 \\ x_4 &= Ax_3 + Bu_3 \\ &= A^3x_1 + A^2Bu_1 + ABu_2 + Bu_3 \\ &\vdots \\ x_T &= A^{T-1}x_1 + A^{T-2}Bu_1 + \cdots + +ABu_{T-2} + Bu_{T-1}. \end{aligned}$$

This can be written in the requested with

$$C = \begin{bmatrix} A^{T-2}B & A^{T-3}B & \cdots & AB & B \end{bmatrix}.$$

The matrix C has size $n \times m(T-1)$.

10.29 *Linear dynamical system with* $2\times$ *down-sampling.* We consider a linear dynamical system with n-vector state x_t, m-vector input u_t, and dynamics given by

$$x_{t+1} = Ax_t + Bu_t, \quad t = 1, 2, \ldots,$$

where A is $n \times n$ matrix A and B is $n \times m$. Define $z_t = x_{2t-1}$ for $t = 1, 2, \ldots$, *i.e.*,

$$z_1 = x_1, \quad z_2 = x_3, \quad z_3 = x_5, \ldots.$$

(The sequence z_t is the original state sequence x_t 'down-sampled' by $2\times$.) Define the $(2m)$-vectors w_t as $w_t = (u_{2t-1}, u_{2t})$ for $t = 1, 2, \ldots$, *i.e.*,

$$w_1 = (u_1, u_2), \quad w_2 = (u_3, u_4), \quad w_3 = (u_5, u_6), \ldots.$$

(Each entry of the sequence w_t is a stack of two consecutive original inputs.) Show that z_t, w_t satisfy the linear dynamics equation $z_{t+1} = Fz_t + Gw_t$, for $t = 1, 2, \ldots$. Give the matrices F and G in terms of A and B.

Solution. Let's work out what z_{t+1} is:

$$\begin{aligned} z_{t+1} &= x_{2t+1} \\ &= Ax_{2t} + Bu_{2t} \\ &= A(Ax_{2t-1} + Bu_{2t-1}) + Bu_{2t} \\ &= A^2x_{2t-1} + ABu_{2t-1} + Bu_{2t} \\ &= A^2z_t + [AB\ B]w_t. \end{aligned}$$

So we see that $F = A^2$ and $G = [AB\ B]$. The matrix F is $n \times n$, and G is $n \times (2m)$.

10.30 *Cycles in a graph.* A *cycle* of length ℓ in a directed graph is a path of length ℓ that starts and ends at the same vertex. Determine the total number of cycles of length $\ell = 10$ for the directed graph given in the example on page 187. Break this number down into the number of cycles that begin (and end) at vertex 1, vertex 2, ..., vertex 5. (These should add up to the total.) *Hint.* Do not count the cycles by hand.

Solution. The matrix $P = A^{10}$ gives the numbers of paths of length 10 that start and end at every vertex. Its diagonal entries give the numbers of paths that start and end at the same vertex, *i.e.*, the numbers of cycles that start (and end) at each vertex. The total number of cycles is the sum of these, *i.e.*, $P_{11} + \cdots + P_{55}$. These numbers give the number of cycles that start at each vertex. They are

$$P_{11} = 53, \quad P_{22} = 42, \quad P_{33} = 58, \quad P_{44} = 35, \quad P_{55} = 20,$$

and the total number of cycles of length 10 is 208.

10.31 *Diameter of a graph.* A directed graph with n vertices is described by its $n \times n$ adjacency matrix A (see §10.3).

(a) Derive an expression P_{ij} for the total number of paths, with length no more than k, from vertex j to vertex i. (We include in this total the number of paths of length zero, which go from each vertex j to itself.) *Hint.* You can derive an expression for the matrix P, in terms of the matrix A.

(b) The *diameter* D of a graph is the smallest number for which there is a path of length $\leq D$ from node j to node i, for every pair of vertices j and i. Using part (a), explain how to compute the diameter of a graph using matrix operations (such as addition, multiplication).

Remark. Suppose the vertices represent all people on earth, and the graph edges represent acquaintance, *i.e.*, $A_{ij} = 1$ if person j and person i are acquainted. (This graph is symmetric.) Even though n is measured in billions, the diameter of this acquaintance graph is thought to be quite small, perhaps 6 or 7. In other words, any two people on earth can be connected though a set of 6 or 7 (or fewer) acquaintances. This idea, originally conjectured in the 1920s, is sometimes called *six degrees of separation*.

Solution.

(a) The number of paths from j to i of length ℓ is given by $(A^\ell)_{ij}$. So the total number of paths of length $\leq \ell$ is P_{ij}, where

$$P = I + A + A^2 + \cdots + A^k.$$

(b) To find the diameter, we evaluate P for $k = 1, 2, \ldots$ and stop the first time P has all entries positive.

10.32 *Matrix exponential.* You may know that for any real number a, the sequence $(1 + a/k)^k$ converges as $k \to \infty$ to the exponential of a, denoted $\exp a$ or e^a. The *matrix exponential* of a square matrix A is defined as the limit of the matrix sequence $(I + A/k)^k$ as $k \to \infty$. (It can shown that this sequence always converges.) The matrix exponential arises in many applications, and is covered in more advanced courses on linear algebra.

(a) Find $\exp 0$ (the zero matrix) and $\exp I$.

(b) Find $\exp A$, for $A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$.

Solution.

(a) $\exp(0) = I$ and $\exp(I) = eI$. More generally, $\exp(aI) = \exp(a)I$.

If $A = 0$, then the matrix $(I + A/k)^k = I$ for all k. Therefore the limit $\exp(A) = I$.

If $A = I$, then the matrix $(I + A/k)^k = (1 + 1/k)^k I$. Therefore the limit $\exp(I) = \exp(1)I$.

If $A = aI$, then the matrix $(I + A/k)^k = (1 + a/k)^k I$. Therefore the limit $\exp(aI) = \exp(a)I$.

(b) $\exp(A) = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$.

We first find the matrix $(I + A/k)^k$. The values of $(I + A/k)^j$ for $j = 1, 2, 3$ are

$$I + A/k = \begin{bmatrix} 1 & 1/k \\ 0 & 1 \end{bmatrix}, \quad (I + A/k)^2 = \begin{bmatrix} 1 & 2/k \\ 0 & 1 \end{bmatrix}, \quad (I + A/k)^3 = \begin{bmatrix} 1 & 3/k \\ 0 & 1 \end{bmatrix}.$$

From this we guess that the general expression is

$$(I + A/k)^j = \begin{bmatrix} 1 & j/k \\ 0 & 1 \end{bmatrix}.$$

This can be proved by induction. The formula is correct for $j = 1$. If it is correct for $j = i$, then

$$(I + A/k)^{i+1} = (I + A/k)^i (I + A/k) = \begin{bmatrix} 1 & i/k \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1/k \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & (i+1)/k \\ 0 & 1 \end{bmatrix}$$

so the formula is also correct for $j = i + 1$. Applying this with $j = k$, we find

$$(I + A/k)^k = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}.$$

10.33 *Matrix equations.* Consider two $m \times n$ matrices A and B. Suppose that for $j = 1, \ldots, n$, the jth column of A is a linear combination of the first j columns of B. How do we express this as a matrix equation? Choose one of the matrix equations below and justify your choice.

(a) $A = GB$ for some upper triangular matrix G.

(b) $A = BH$ for some upper triangular matrix H.

(c) $A = FB$ for some lower triangular matrix F.

(d) $A = BJ$ for some lower triangular matrix J.

Solution. The correct answer is (b).
To see this, let a_j and b_j be the jth columns of A and B. We are given that a_j can be expressed as a linear combination of the columns $b_1, \ldots, b_j$:

$$a_j = \beta_{1j} b_1 + \beta_{2j} b_2 + \cdots + \beta_{jj} b_j = B \begin{bmatrix} \beta_{1j} \\ \beta_{2j} \\ \vdots \\ \beta_{jj} \end{bmatrix}$$

for some coefficients $\beta_{1j}, \ldots, \beta_{jj}$. Therefore

$$A = \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix} = B \begin{bmatrix} \beta_{11} & \beta_{12} & \cdots & \beta_{1n} \\ 0 & \beta_{22} & \cdots & \beta_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \beta_{mn} \end{bmatrix}.$$

10.34 Choose one of the responses *always*, *never*, or *sometimes* for each of the statements below. 'Always' means the statement is always true, 'never' means it is never true, and 'Sometimes' means it can be true or false, depending on the particular values of the matrix or matrices. Give a brief justification of each answer.

(a) An upper triangular matrix has linearly independent columns.

(b) The rows of a tall matrix are linearly dependent.

(c) The columns of A are linearly independent, and $AB = 0$ for some nonzero matrix B.

Solution.

(a) *Sometimes.* The zero matrix is upper triangular and has linearly dependent columns. The identity matrix is upper triangular and has linearly independent columns.

(b) *Always.* This is true by the independence-dimension inequality.

(c) *Never.* If $AB = 0$ for some nonzero matrix B, then B has some column that is nonzero, say, the kth column b_k. $AB = 0$ implies $Ab_k = 0$ (since this is the kth column of AB). But $Ab_k = 0$ with $b_k \neq 0$ means that the columns of A are linearly dependent.

10.35 *Orthogonal matrices.* Let U and V be two orthogonal $n \times n$ matrices. Show that the matrix UV and the $(2n) \times (2n)$ matrix

$$\frac{1}{\sqrt{2}} \begin{bmatrix} U & U \\ V & -V \end{bmatrix}$$

are orthogonal.

Solution.

(a) The product UV is a square matrix that satisfies

$$(UV)^T UV = V^T U^T U V^T = V^T I V = V^T V = I.$$

(b) The matrix is square and satisfies

$$\begin{aligned} \left(\frac{1}{\sqrt{2}} \begin{bmatrix} U & U \\ V & -V \end{bmatrix}\right)^T \left(\frac{1}{\sqrt{2}} \begin{bmatrix} U & U \\ V & -V \end{bmatrix}\right) &= \frac{1}{2} \begin{bmatrix} U^T & V^T \\ U^T & -V^T \end{bmatrix} \begin{bmatrix} U & U \\ V & -V \end{bmatrix} \\ &= \frac{1}{2} \begin{bmatrix} U^T U + V^T V & U^T U - V^T V \\ U^T U - V^T V & U^T U + V^T V \end{bmatrix} \\ &= \frac{1}{2} \begin{bmatrix} 2I_n & 0 \\ 0 & 2I_n \end{bmatrix} \\ &= I_{2n}. \end{aligned}$$

10.36 *Quadratic form.* Suppose A is an $n \times n$ matrix and x is an n-vector. The triple product $x^T A x$, a 1×1 matrix which we consider to be a scalar (*i.e.*, number), is called a *quadratic form* of the vector x, with coefficient matrix A. A quadratic form is the vector analog of a quadratic function αu^2, where α and u are both numbers. Quadratic forms arise in many fields and applications.

(a) Show that $x^T A x = \sum_{i,j=1}^n A_{ij} x_i x_j$.

(b) Show that $x^T (A^T) x = x^T A x$. In other words, the quadratic form with the transposed coefficient matrix has the same value for any x. *Hint.* Take the transpose of the triple product $x^T A x$.

(c) Show that $x^T ((A + A^T)/2) x = x^T A x$. In other words, the quadratic form with coefficient matrix equal to the symmetric part of a matrix (*i.e.*, $(A + A^T)/2$) has the same value as the original quadratic form.

(d) Express $2x_1^2 - 3x_1 x_2 - x_2^2$ as a quadratic form, with symmetric coefficient matrix A.

10.37 *Orthogonal 2×2 matrices.* In this problem, you will show that every 2×2 orthogonal matrix is either a rotation or a reflection (see §7.1).

(a) Let

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

be an orthogonal 2×2 matrix. Show that the following equations hold:

$$a^2 + c^2 = 1, \qquad b^2 + d^2 = 1, \qquad ab + cd = 0.$$

(b) Define $s = ad - bc$. Combine the three equalities in part (a) to show that

$$|s| = 1, \qquad b = -sc, \qquad d = sa.$$

(c) Suppose $a = \cos\theta$. Show that there are two possible matrices Q: A rotation (counterclockwise over θ radians), and a reflection (through the line that passes through the origin at an angle of $\theta/2$ radians with respect to horizontal).

Solution.

(a) Since Q is orthogonal, $I = Q^TQ$. This gives us

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} a & c \\ b & d \end{bmatrix} \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} a^2+c^2 & ab+cd \\ ba+dc & b^2+d^2 \end{bmatrix}.$$

This gives the three equalities.

(b) Combining the identities in part (a) gives

$$\begin{aligned} s^2 &= (ad-bc)^2 \\ &= a^2d^2 + c^2b^2 - abcd - abcd \\ &= a^2d^2 + c^2b^2 + a^2b^2 + c^2d^2 \\ &= (a^2+c^2)(b^2+d^2) \\ &= 1. \end{aligned}$$

On line 3 we used $ab = -cd$. The other two identities now follow directly from part (a):

$$-sc = -(ad-bc)c = -adc + bc^2 = (a^2+c^2)b = b.$$

and

$$sa = (ad-bc)a = a^2d - abc = (a^2+c^2)d = d$$

(c) From the first equality in part (a), $c = \pm\sqrt{1-a^2} = \pm\sin\theta$. In part (b), we can have have $s = \pm 1$. There are four combinations of the two signs:

$$Q_1 = \begin{bmatrix} \cos\theta & \sin\theta \\ \sin\theta & -\cos\theta \end{bmatrix}, \qquad Q_2 = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix},$$

$$Q_3 = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & -\cos\theta \end{bmatrix}, \qquad Q_4 = \begin{bmatrix} \cos\theta & -\sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}.$$

If $\cos\theta\sin\theta \neq 0$, the matrices Q_3 and Q_4 are not orthogonal. (They do not satisfy the third equality in part (a).)

If $\cos\theta\sin\theta = 0$, the matrices Q_3 and Q_4 have the same form as Q_1 and Q_2. If $\sin\theta = 0$, then $Q_3 = Q_1$ and $Q_4 = Q_2$. If $\cos\theta = 0$, then Q_3 and Q_4 have the same form as Q_2 and Q_1, respectively, if we flip the sign of θ.

The matrices Q_1 and Q_2 are orthogonal. Q_1 is a reflection matrix. Q_2 is a rotation.

10.38 *Orthogonal matrix with nonnegative entries.* Suppose the $n \times n$ matrix A is orthogonal, and all of its entries are nonnegative, *i.e.*, $A_{ij} \geq 0$ for $i, j = 1, \ldots, n$. Show that A must be a permutation matrix, *i.e.*, each entry is either 0 or 1, each row has exactly one entry with value one, and each column has exactly one entry with value one. (See page 132.)

Solution. In fact, the matrix A must be a permutation matrix, *i.e.*, all entries are either 0 or 1, and each row has exactly one entry with value one, and each column has exactly one entry with value one.

Let's see how we came to this conclusion. First of all, the columns $a_1, \ldots, a_n$ of A are orthogonal to each other. But if two nonnegative vectors are orthogonal, it means that for each entry, if one is nonzero the other one must be zero. For each i, one of the columns a_j must be nonzero; otherwise the matrix A has a row of all zeros, and then A isn't invertible. But when the column a_k has a nonzero entry in the ith position, it implies that all the other columns have zero in the ith position. This means that for each row of A, there is exactly one entry that is nonzero. But the norm of the rows is one, so the sum of the squares of the entries is 1; this means that the value of the one nonzero entry is 1 or -1. Since the entries are nonnegative, it must be one.

10.39 *Gram matrix and QR factorization.* Suppose the matrix A has linearly independent columns and QR factorization $A = QR$. What is the relationship between the Gram matrix of A and the Gram matrix of R? What can you say about the angles between the columns of A and the angles between the columns of R?

Solution. They are the same. To see this, we note that $A = QR$ so

$$G = A^TA = (QR)^T(QR) = R^TQ^TQR = R^T(Q^TQ)R = R^TR.$$

So G is the Gram matrix of A and also of R.

This implies that the angles between the columns of A are the same as the angles between the corresponding columns of R: $\angle(a_i, a_j) = \angle(r_i, r_j)$, where a_i is the ith column of A and r_i is the ith column of R.

10.40 *QR factorization of first i columns of A.* Suppose the $n \times k$ matrix A has QR factorization $A = QR$. We define the $n \times i$ matrices

$$A_i = \begin{bmatrix} a_1 & \cdots & a_i \end{bmatrix}, \qquad Q_i = \begin{bmatrix} q_1 & \cdots & q_i \end{bmatrix},$$

for $i = 1, \ldots, k$. Define the $i \times i$ matrix R_i as the submatrix of R containing its first i rows and columns, for $i = 1, \ldots, k$. Using index range notation, we have

$$A_i = A_{1:n,1:i}, \quad Q_i = A_{1:n,1:i}, \quad R_i = R_{1:i,1:i}.$$

Show that $A_i = Q_iR_i$ is the QR factorization of A_i. This means that when you compute the QR factorization of A, you are also computing the QR factorization of all submatrices $A_1, \ldots, A_k$.

Solution. Using block matrix notation we can write the QR factorization of A as

$$\begin{bmatrix} A_i & A_{:,i+1:k} \end{bmatrix} = \begin{bmatrix} Q_i & Q_{:,i+1:k} \end{bmatrix} \begin{bmatrix} R_i & R_{1:i,i+1:k} \\ 0 & R_{i+1:k,i+1:k} \end{bmatrix}.$$

The $2,1$ block on R is zero because R is upper triangular. Multiplying out the matrices on the right gives

$$\begin{bmatrix} A_i & A_{:,i+1:k} \end{bmatrix} = \begin{bmatrix} Q_iR_i & Q_iR_{1:i,i+1:k} + Q_{:,i+1:k} \end{bmatrix}.$$

The first block column is $A_i = Q_iR_i$. This is the QR factorization of A_i because Q_i has orthonormal columns and R_i is upper triangular with positive diagonal entries.

10.41 *Clustering via k-means as an approximate matrix factorization.* Suppose we run the k-means algorithm on the N n-vectors $x_1, \ldots, x_N$, to obtain the group representatives $z_1, \ldots, z_k$. Define the matrices

$$X = \begin{bmatrix} x_1 & \cdots & x_N \end{bmatrix}, \qquad Z = \begin{bmatrix} z_1 & \cdots & z_N \end{bmatrix}.$$

X has size $n \times N$ and Z has size $n \times k$. We encode the assignment of vectors to groups by the $k \times N$ clustering matrix C, with $C_{ij} = 1$ if x_j is assigned to group i, and $C_{ij} = 0$ otherwise. Each column of C is a unit vector; its transpose is a selector matrix.

(a) Give an interpretation of the columns of the matrix $X - ZC$, and the squared norm (matrix) norm $\|X - ZC\|^2$.

(b) Justify the following statement: The goal of the k-means algorithm is to find an $n \times k$ matrix Z, and a $k \times N$ matrix C, which is the transpose of a selector matrix, so that $\|X - ZC\|$ is small, *i.e.*, $X \approx ZC$.

Solution.

(a) Using the notation c_i for the cluster to which vector i is assigned (see §4.2), we can write the matrix ZC as

$$ZC = \begin{bmatrix} z_{c_1} & z_{c_2} & \cdots & z_{c_N} \end{bmatrix}$$

Therefore

$$X - ZC = \begin{bmatrix} x_1 - z_{c_1} & x_2 - z_{c_2} & \cdots & x_N - z_{c_N} \end{bmatrix}$$

Column i is the difference of the vector x_i and its group representative. Therefore

$$\|X - ZC\|^2 = \sum_{i=1}^{N} \|x_i - z_{c_i}\|^2 = NJ^{\text{clust}}.$$

(b) Minimizing the clustering objective J^{clust} is the same as minimizing $\|X - ZC\|^2$, over all possible $n \times k$ matrices Z and all possible transposed selector matrices C.

10.42 A matrix-vector multiplication Ax of an $n \times n$ matrix A and an n-vector x takes $2n^2$ flops in general. Formulate a faster method, with complexity linear in n, for matrix-vector multiplication with the matrix $A = I + ab^T$, where a and b are given n-vectors.

Solution. We write the product $y = Ax$ as

$$y = (I + ab^T)x = x + a(b^T x).$$

This can be computed by first computing the inner product $\gamma = b^T x$ ($2n - 1$ flops), then the scalar multiplication $z = \gamma a$ (n flops), and finally the addition $y = x + z$ (n flops). The total is $4n - 2 \approx 4n$ flops.

10.43 A particular computer takes about 0.2 seconds to multiply two 1500×1500 matrices. About how long would you guess the computer takes to multiply two 3000×3000 matrices? Give your prediction (*i.e.*, the time in seconds), and your (very brief) reasoning.

Solution. Multiplying $n \times n$ matrices requires order n^3 flops. If you double n, the number of flops goes up by a factor of 8. So we'd guess the time would be something like $8\times$ more, around 1.6 seconds. (In fact it takes around 1.3 seconds.)

10.44 *Complexity of matrix quadruple product.* (See page 182.) We wish to compute the product $E = ABCD$, where A is $m \times n$, B is $n \times p$, C is $p \times q$, and D is $q \times r$.

(a) Find all methods for computing E using three matrix-matrix multiplications. For example, you can compute AB, CD, and then the product $(AB)(CD)$. Give the total number of flops required for each of these methods. *Hint.* There are four other methods.

(b) Which method requires the fewest flops, with dimensions $m = 10$, $n = 1000$, $p = 10$, $q = 1000$, $r = 100$?

Solution. There are five ways to compute E.

- First compute AB and CD, then their product. This requires $2mnp + 2pqr + 2mpr$ flops.
- Compute AB, then $(AB)C$, and finally $((AB)C)D$. This requires $2mnp + 2mpq + 2mqr$ flops.
- Compute BC, then $A(BC)$, and finally $(A(BC))D$. This requires $2npq + 2mnq + 2mqr$ flops.
- Compute BC, then $(BC)D$, and finally $A((BC)D)$. This requires $2npq + 2nqr + 2mnr$ flops.
- Compute CD, then $B(CD)$, and finally $A(B(CD))$. This requires $2pqr + 2npr + 2mnr$ flops.

Chapter 11

Matrix inverses

Exercises

11.1 *Affine combinations of left inverses.* Let Z be a tall $m \times n$ matrix with linearly independent columns, and let X and Y be left inverses of Z. Show that for any scalars α and β satisfying $\alpha + \beta = 1$, $\alpha X + \beta Y$ is also a left inverse of Z. It follows that if a matrix has two different left inverses, it has an infinite number of different left inverses.

Solution. We are asked to show that $(\alpha X + \beta Y)Z = I$.

$$(\alpha X + \beta Y)Z = \alpha XZ + \beta YZ = \alpha I + \beta I = (\alpha + \beta)I = I.$$

11.2 *Left and right inverses of a vector.* Suppose that x is a nonzero n-vector with $n > 1$.

(a) Does x have a left inverse?

(b) Does x have a right inverse?

In each case, if the answer is yes, give a left or right inverse; if the answer is no, give a specific nonzero vector and show that it is not left- or right-invertible.

Solution. Yes, x has a left inverse, in fact, many left inverses. Here's a simple one, which is the pseudo-inverse:

$$x^\dagger = (x^T x)^{-1} x^T = (1/\|x\|^2) x^T.$$

You can check that $x^\dagger x = 1$.

No, x does not have a right inverse. To have a right inverse its rows must be linearly independent, but since x is tall, that's not possible.

11.3 *Matrix cancellation.* Suppose the scalars a, x, and y satisfy $ax = ay$. When $a \neq 0$ we can conclude that $x = y$; that is, we can cancel the a on the left of the equation. In this exercise we explore the matrix analog of cancellation, specifically, what properties of A are needed to conclude $X = Y$ from $AX = AY$, for matrices A, X, and Y?

(a) Give an example showing that $A \neq 0$ is not enough to conclude that $X = Y$.

(b) Show that if A is left-invertible, we can conclude from $AX = AY$ that $X = Y$.

(c) Show that if A is not left-invertible, there are matrices X and Y with $X \neq Y$, and $AX = AY$.

Remark. Parts (b) and (c) show that you can cancel a matrix on the left when, and only when, the matrix is left-invertible.

Solution. Here is an

(a) Take $A = (1, 0)$, $X = (0, 1)$, $Y = (0, 0)$ (*i.e.*, they are all 2×1 matrices.

(b) Let B be a left inverse of A. Multiply $AX = AY$ on the left by B to get $B(AX) = B(AY)$. Re-associate as $(BA)X = (BA)Y$. From $BA = I$ we conclude that $X = Y$.

(c) Now suppose that A is not left-invertible. This means that its columns are linearly dependent. This means that there is a nonzero vector x for which $Ax = 0$. Then we have $Ax = Ay$, with $y = 0$, but $x \neq y$. This is a counterexample, where we simply consider x and y as $n \times 1$ matrices.

11.4 *Transpose of orthogonal matrix.* Let U be an orthogonal $n \times n$ matrix. Show that its transpose U^T is also orthogonal.

Solution. If U is square and $U^T U = I$, then U^T is the inverse of U and $UU^T = I$. Since U is the transpose of U^T, this shows that the matrix U^T is orthogonal.

11.5 *Inverse of a block matrix.* Consider the $(n + 1) \times (n + 1)$ matrix

$$A = \begin{bmatrix} I & a \\ a^T & 0 \end{bmatrix},$$

where a is an n-vector.

(a) When is A invertible? Give your answer in terms of a. Justify your answer.

(b) Assuming the condition you found in part (a) holds, give an expression for the inverse matrix A^{-1}.

Solution.

(a) A is invertible if and only if $a \neq 0$.

We use the property that a square matrix is invertible if and only if its columns are linearly independent, *i.e.*, $Ax = 0$ implies $x = 0$. We partition x as $x = (y, z)$ with y an n-vector and z scalar. Then

$$Ax = \begin{bmatrix} I & a \\ a^T & 0 \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix} = \begin{bmatrix} y + az \\ a^T x \end{bmatrix}.$$

Therefore $Ax = 0$ is equivalent to

$$y = -za, \qquad a^T y = 0.$$

If $a = 0$, this holds with $y = 0$, $z = 1$. Therefore A is not invertible if $a = 0$. Suppose $a \neq 0$. Substituting $y = -za$ in the second equation gives $-za^T a = 0$. Since $a \neq 0$ this implies $z = 0$. If $z = 0$ then $y = -za = 0$. This shows that A is invertible.

(b) The inverse is given by

$$A^{-1} = \frac{1}{\|a\|^2} \begin{bmatrix} \|a\|^2 I - aa^T & a \\ a^T & -1 \end{bmatrix}.$$

We can check that this is the inverse by simply multiplying A on the left or on the right by our block matrix and confirming the product is the identity. You don't need to give any more justification than this; if you show that the product of two matrices is the identity, then they are inverses.

In any case, let's see how to *derive* this inverse. To do this, we'll solve the linear equations $Ax = b$, and equate the result with $A^{-1}b$. First partition the two vectors as $x = (x_1, x_2)$, $d = (b_1, b_2)$, with x_1 and b_1 n-vectors, and x_2 and b_2 scalars. The equation $Ax = b$ can be written as

$$\begin{bmatrix} I & a \\ a^T & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}.$$

This gives two equations

$$x_1 + ax_2 = b_1, \qquad a^T x_1 = b_2,$$

where we write ax_2 as $x_2 a$ since x_2 is a scalar. From the first equation we get $x_1 = b_1 - x_2 a$; substituting this into the second equation we get

$$b_2 = a^T(b_1 - x_2 a) = a^T b_1 - x_2 \|a\|^2.$$

Solving for x_2 and substituting the result in $x_1 = b_1 - x_2 a$ gives so

$$x_2 = (a^T b_1 - b_2)/\|a\|^2, \qquad x_1 = b_1 - ((a^T b_1 - b_2)/\|a\|^2)a.$$

Writing this out in matrix form we get

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \frac{1}{\|a\|^2} \begin{bmatrix} \|a\|^2 I - aa^T & a \\ a^T & -1 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}.$$

This holds for any b, so the matrix on the right must be equal to the inverse of A.

11.6 *Inverse of a block upper triangular matrix.* Let B and D be invertible matrices of sizes $m \times m$ and $n \times n$, respectively, and let C be any $m \times n$ matrix. Find the inverse of

$$A = \begin{bmatrix} B & C \\ 0 & D \end{bmatrix}$$

in terms of B^{-1}, C, and D^{-1}. (The matrix A is called *block upper triangular*.)

Hints. First get an idea of what the solution should look like by considering the case when B, C, and D are scalars. For the matrix case, your goal is to find matrices W, X, Y, Z (in terms of B^{-1}, C, and D^{-1}) that satisfy

$$A \begin{bmatrix} W & X \\ Y & Z \end{bmatrix} = I.$$

Use block matrix multiplication to express this as a set of four matrix equations that you can then solve. The method you will find is sometimes called *block back substitution*.

Solution. We determine W, X, Y, Z from the equation

$$\begin{bmatrix} B & C \\ 0 & D \end{bmatrix} \begin{bmatrix} W & X \\ Y & Z \end{bmatrix} = \begin{bmatrix} I_m & 0 \\ 0 & I_n \end{bmatrix}.$$

This is equivalent to the four equations

$$BW + CY = I, \qquad DY = 0, \qquad BX + CZ = 0, \qquad DZ = I.$$

The equation $DY = 0$ implies that $Y = 0$ because D is invertible. The first equation then reduces to $BW = I$, which gives $W = B^{-1}$. From the last equation we see that $Z = D^{-1}$. The third equation reduces to $BX = -CD^{-1}$ which gives $X = -B^{-1}CD^{-1}$. We conclude that the inverse is

$$\begin{bmatrix} W & X \\ Y & Z \end{bmatrix} = \begin{bmatrix} B^{-1} & -B^{-1}CD^{-1} \\ 0 & D^{-1} \end{bmatrix}.$$

11.7 *Inverse of an upper triangular matrix.* Suppose the $n \times n$ matrix R is upper triangular and invertible, *i.e.*, its diagonal entries are all nonzero. Show that R^{-1} is also upper triangular. *Hint.* Use back substitution to solve $Rs_k = e_k$, for $k = 1, \ldots, n$, and argue that $(s_k)_i = 0$ for $i > k$.

Solution. The kth column of $S = R^{-1}$ is $s_k = R^{-1}e_k$. So arguing that $(s_k)_i = 0$ for $i > k$ is the same as showing that R^{-1} is upper triangular.

To see why $(s_k)_i = 0$ for $i > k$, we look at what happens when we apply back substitution with right-hand side $b = e_k$. Assuming $k < n$, in the first step, $(s_k)_n = b_n/R_{nn} = 0$. In the second and subsequent steps, we also have $x_i = 0$, until we get to index k, and then we have $(s_k)_k = 1/R_{kk}$. This shows that R^{-1} is lower triangular.

11.8 *If a matrix is small, its inverse is large.* If a number a is small, its inverse $1/a$ (assuming $a \neq 0$) is large. In this exercise you will explore a matrix analog of this idea. Suppose the $n \times n$ matrix A is invertible. Show that $\|A^{-1}\| \geq \sqrt{n}/\|A\|$. This implies that if a matrix is small, its inverse is large. *Hint.* You can use the inequality $\|AB\| \leq \|A\|\|B\|$, which holds for any matrices for which the product makes sense. (See exercise 10.12.)

Solution. The result follows from the norm inequality $\|AB\| \leq \|A\|\|B\|$. Applying this with $B = A^{-1}$, we get $\sqrt{n} \leq \|A\|\|A^{-1}\|$, since the norm of the $n \times n$ identity matrix is $\sqrt{n}$.

11.9 *Push-through identity.* Suppose A is $m \times n$, B is $n \times m$, and the $m \times m$ matrix $I + AB$ is invertible.

(a) Show that the $n \times n$ matrix $I + BA$ is invertible. *Hint.* Show that $(I + BA)x = 0$ implies $(I + AB)y = 0$, where $y = Ax$.

(b) Establish the identity

$$B(I + AB)^{-1} = (I + BA)^{-1}B.$$

This is sometimes called the *push-through identity* since the matrix B appearing on the left 'moves' into the inverse, and 'pushes' the B in the inverse out to the right side. *Hint.* Start with the identity

$$B(I + AB) = (I + BA)B,$$

and multiply on the right by $(I + AB)^{-1}$, and on the left by $(I + BA)^{-1}$.

Solution.

(a) Suppose $(I + BA)x = 0$. We show that $x = 0$.
Multiplying on the left with A gives $A(I + BA)x = 0$. This can be written as

$$A(I + BA)x = Ax + ABAx = (I + AB)Ax = 0.$$

Since the matrix $I + AB$ is invertible, this implies that $Ax = 0$. Combining this with our assumption that $0 = (I + BA)x = x + BAx$, we get $x = 0$.

(b) The identity $B(I + AB) = (I + BA)B$ is obvious, since both sides of the equation are equal to $B + BAB$. Following the hint, we get

$$(I + BA)^{-1}B(I + AB)(I + AB)^{-1} = (I + BA)^{-1}(I + BA)B(I + AB)^{-1},$$

which simplifies to the push-through identity $(I + BA)^{-1}B = B(I + AB)^{-1}$.

11.10 *Reverse-time linear dynamical system.* A linear dynamical system has the form

$$x_{t+1} = Ax_t,$$

where x_t in the (n-vector) state in period t, and A is the $n \times n$ dynamics matrix. This formula gives the state in the next period as a function of the current state.
We want to derive a recursion of the form

$$x_{t-1} = A^{\text{rev}}x_t,$$

which gives the previous state as a function of the current state. We call this the *reverse time linear dynamical system*.

(a) When is this possible? When it is possible, what is A^{rev}?

(b) For the specific linear dynamical system with dynamics matrix

$$A = \begin{bmatrix} 3 & 2 \\ -1 & 4 \end{bmatrix},$$

find A^{rev}, or explain why the reverse time linear dynamical system doesn't exist.

Solution.

(a) This is possible if and only if A is invertible, and when it is invertible, $A^{\text{rev}} = A^{-1}$.

(b) By using the formula for 2×2 matrix inverse, or manually solving a system of linear equations, we get

$$A^{\text{rev}} = A^{-1} = \frac{1}{14}\begin{bmatrix} 4 & -2 \\ 1 & 3 \end{bmatrix}.$$

11.11 *Interpolation of rational functions.* (Continuation of exercise 8.8.) Find a rational function

$$f(t) = \frac{c_1 + c_2 t + c_3 t^2}{1 + d_1 t + d_2 t^2}$$

that satisfies the following interpolation conditions:

$$f(1) = 2, \qquad f(2) = 5, \qquad f(3) = 9, \qquad f(4) = -1, \qquad f(5) = -4.$$

In exercise 8.8 these conditions were expressed as a set of linear equations in the coefficients c_1, c_2, c_3, d_1 and d_2; here we are asking you to form and (numerically) solve the system of equations. Plot the rational function you find over the range $x = 0$ to $x = 6$. Your plot should include markers at the interpolation points $(1, 2), \ldots, (5, -4)$. (Your rational function graph should pass through these points.)

Solution. We write the five interpolation conditions as

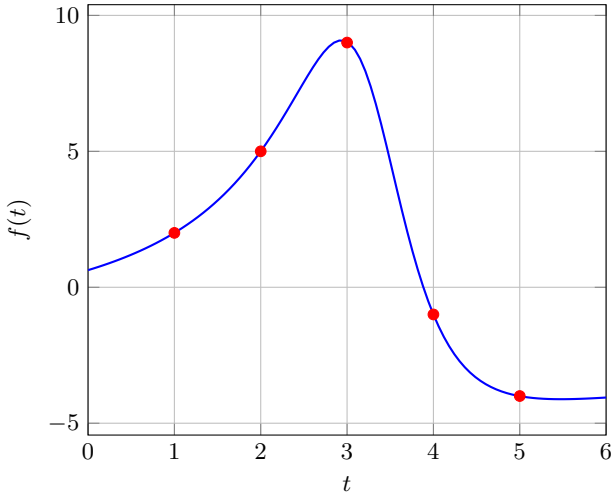
$$\begin{aligned}
c_1 + c_2 + c_3 &= 2(1 + d_1 + d_2) \\
c_1 + 2c_2 + 4c_3 &= 5(1 + 2d_1 + 4d_2) \\
c_1 + 3c_2 + 9c_3 &= 9(1 + 3d_1 + 9d_2) \\
c_1 + 4c_2 + 16c_3 &= -(1 + 4d_1 + 16d_2) \\
c_1 + 5c_2 + 25c_3 &= -4(1 + 5d_1 + 25d_2).
\end{aligned}$$

This is a set of linear equations in five variables

$$\begin{bmatrix} 1 & 1 & 1 & -2 & -2 \\ 1 & 2 & 4 & -10 & -20 \\ 1 & 3 & 9 & -27 & -81 \\ 1 & 4 & 16 & 4 & 16 \\ 1 & 5 & 25 & 20 & 100 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ d_1 \\ d_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \\ 9 \\ -1 \\ -4 \end{bmatrix}.$$

The solution is

$$c_1 = 0.6296, \quad c_2 = 0.6049, \quad c_3 = -0.1975, \quad d_1 = -0.5679, \quad d_2 = 0.0864.$$



11.12 *Combinations of invertible matrices.* Suppose the $n \times n$ matrices A and B are both invertible. Determine whether each of the matrices given below is invertible, without any further assumptions about A and B.

(a) $A + B$.

(b) $\begin{bmatrix} A & 0 \\ 0 & B \end{bmatrix}$.

(c) $\begin{bmatrix} A & A+B \\ 0 & B \end{bmatrix}$.

(d) ABA.

Solution.

(a) False. Take $A = -B$. Then $A + B = 0$, which is not invertible.

(b) True.

$$\begin{bmatrix} A & 0 \\ 0 & B \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} & 0 \\ 0 & B^{-1} \end{bmatrix}.$$

(c) True.

$$\begin{bmatrix} A & A+B \\ 0 & B \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} & -A^{-1} - B^{-1} \\ 0 & B^{-1} \end{bmatrix}.$$

(d) True. $(ABA)^{-1} = A^{-1}B^{-1}A^{-1}$.

11.13 *Another left inverse.* Suppose the $m \times n$ matrix A is tall and has linearly independent columns. One left inverse of A is the pseudo-inverse $A^\dagger$. In this problem we explore another one. Write A as the block matrix

$$A = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix},$$

where A_1 is $n \times n$. We assume that A_1 is invertible (which need not happen in general). Show that the following matrix is a left inverse of A:

$$\tilde{A} = \begin{bmatrix} A_1^{-1} & 0_{n\times(m-n)} \end{bmatrix}.$$

Solution. We verify that $\tilde{A}A = I$.

$$\tilde{A}A = \begin{bmatrix} A_1^{-1} & 0_{n\times(m-n)} \end{bmatrix} \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} = A_1^{-1}A_1 = I.$$

11.14 *Middle inverse.* Suppose A is an $n \times p$ matrix and B is a $q \times n$ matrix. If a $p \times q$ matrix X exists that satisfies $AXB = I$, we call it a *middle inverse* of the pair A, B. (This is not a standard concept.) Note that when A or B is an identity matrix, the middle inverse reduces to the right or left inverse, respectively.

(a) Describe the conditions on A and B under which a middle inverse X exists. Give your answer using only the following four concepts: Linear independence of the rows or columns of A, and linear independence of the rows or columns of B. You must justify your answer.

(b) Give an expression for a middle inverse, assuming the conditions in part (a) hold.

Solution.

(a) The matrix A must have linearly independent rows and B must have linearly independent columns.

To see this, suppose $AXB = I$. Then XB is a right inverse of A, so A is right-invertible, which implies that A has linearly independent rows. Similarly, AX is a

left inverse of B, so B is left-invertible, which implies that B has linearly independent columns.

The converse is also true: If A has linearly independent rows and B has linearly independent columns, then there exists an X for which $AXB = I$. In this case A has a right inverse, say, C, and B has a left inverse, say, D. Define $X = CD$. Then we have $AXB = A(CD)B = (AC)(DB) = I$.

(b) A specific middle-inverse is

$$X = A^\dagger B^\dagger = A^T(AA^T)^{-1}(B^TB)^{-1}B^T.$$

11.15 *Invertibility of population dynamics matrix.* Consider the population dynamics matrix

$$A = \begin{bmatrix} b_1 & b_2 & \cdots & b_{99} & b_{100} \\ 1-d_1 & 0 & \cdots & 0 & 0 \\ 0 & 1-d_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1-d_{99} & 0 \end{bmatrix},$$

where $b_i \geq 0$ are the birth rates and $0 \leq d_i \leq 1$ are death rates. What are the conditions on b_i and d_i under which A is invertible? (If the matrix is never invertible or always invertible, say so.) Justify your answer.

Solution. The conditions are that $d_i < 1$ for $i = 1, \ldots, 99$, and $b_{100} > 0$.

First suppose the conditions do not hold. If $d_k = 1$ for some k, then the $(k+1)$th row of A is zero, so the rows of A are linearly dependent and A is not invertible. If $b_{100} = 0$, then the last column of A is zero, so the columns of A are linearly dependent and A is not invertible.

Now we show the converse. Suppose that $d_i < 1$ and $b_{100} > 0$. We show that $Ax = 0$ implies $x = 0$. The product Ax is

$$Ax = \begin{bmatrix} b_1x_1 + b_2x_2 + \cdots + b_{100}x_{100} \\ (1-d_1)x_1 \\ \vdots \\ (1-d_{99})x_{99} \end{bmatrix}.$$

Suppose $Ax = 0$. Then $(1-d_i)x_i = 0$ for $i = 1, \ldots, 99$ and, since $d_i < 1$, we have $x_1 = \cdots = x_{99} = 0$. The first entry of Ax then reduces to $b_{100}x_{100}$ and, since $b_{100} \neq 0$, we have $x_{100} = 0$.

11.16 *Inverse of running sum matrix.* Find the inverse of the $n \times n$ running sum matrix,

$$S = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ 1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 1 & \cdots & 1 & 0 \\ 1 & 1 & \cdots & 1 & 1 \end{bmatrix}.$$

Does your answer make sense?

Solution. The inverse is

$$S^{-1} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ -1 & 1 & \cdots & 0 & 0 \\ & & \ddots & & \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & -1 & 1 \end{bmatrix}.$$

The matrix S is lower triangular. To find the inverse, we solve $SX = I$, column by column, by forward substitution. The equations for column j are

$$\begin{aligned} X_{1j} &= 0 \\ X_{1j} + X_{2j} &= 0 \\ &\vdots \\ X_{1j} + X_{2j} + \cdots + X_{j-1,j} &= 0 \\ X_{1j} + X_{2j} + \cdots + X_{jj} &= 1 \\ X_{1j} + X_{2j} + \cdots + X_{j+1,j} &= 0 \\ &\vdots \\ X_{1j} + X_{2j} + \cdots + X_{nj} &= 0. \end{aligned}$$

If $j < n$, then forward substitution gives

$$X_{1j} = \cdots = X_{j-1,j} = 0, \qquad X_{jj} = 1, \quad X_{j+1,j} = -1, \qquad X_{j+2,j} = \cdots = X_{nj} = 0.$$

If $j = n$, the solution is $X_{1n} = \cdots = X_{n-1,n} = 0$ and $X_{nn} = 1$.

The matrix S^{-1} is the difference matrix, with an additional top row that is e_1^T. We have

$$S^{-1}y = (y_1, y_2 - y_1, \ldots, y_n - y_{n-1}).$$

This makes sense: to undo a running sum (which is like an integral), you should differentiate, which is what S^{-1} (sort of) does.

11.17 *A matrix identity.* Suppose A is a square matrix that satisfies $A^k = 0$ for some integer k. (Such a matrix is called *nilpotent*.) A student guesses that $(I-A)^{-1} = I+A+\cdots+A^{k-1}$, based on the infinite series $1/(1-a) = 1 + a + a^2 + \cdots$, which holds for numbers a that satisfy $|a| < 1$.

Is the student right or wrong? If right, show that her assertion holds with no further assumptions about A. If she is wrong, give a counterexample, *i.e.*, a matrix A that satisfies $A^k = 0$, but $I + A + \cdots + A^{k-1}$ is not the inverse of $I - A$.

Solution. The student is right: the matrix $I - A$ is invertible and its inverse is $I + A + \cdots + A^{k-1}$. To see this:

$$\begin{aligned} (I - A)(I + A + \cdots + A^{k-1}) &= I + A + \cdots + A^{k-1} - (A + A^2 + \cdots + A^k) \\ &= I + A^k \\ &= I. \end{aligned}$$

In the last step we use the property that A is nilpotent.

11.18 *Tall-wide product.* Suppose A is an $n \times p$ matrix and B is a $p \times n$ matrix, so $C = AB$ makes sense. Explain why C cannot be invertible if A is tall and B is wide, *i.e.*, if $p < n$. *Hint.* First argue that the columns of B must be linearly dependent.

Solution. By the independence-dimension inequality, the columns of B are linearly dependent, since they have dimension p and there are n of them, and $n > p$. So there is a nonzero n-vector x that satisfies $Bx = 0$. This implies that $(AB)x = A(Bx) = A0 = 0$. This shows that the columns of $C = AB$ are linearly dependent, and so C is not invertible.

11.19 *Control restricted to one time period.* A linear dynamical system has the form $x_{t+1} = Ax_t + u_t$, where the n-vector x_t is the state and u_t is the input at time t. Our goal is to choose the input sequence $u_1, \ldots, u_{N-1}$ so as to achieve $x_N = x^{\mathrm{des}}$, where x^{des} is a given n-vector, and N is given. The input sequence must satisfy $u_t = 0$ unless $t = K$, where $K < N$ is given. In other words, the input can only act at time $t = K$. Give a formula for u_K that achieves this goal. Your formula can involve A, N, K, x_1, and x^{des}. You

can assume that A is invertible. *Hint.* First derive an expression for x_K, then use the dynamics equation to find x_{K+1}. From x_{K+1} you can find x_N.

Solution. For $t \leq K$ we have $u_t = 0$, *i.e.*, $x_{t+1} = Ax_t$, so the state is $x_t = A^{t-1}x_1$. In particular $x_K = A^{K-1}x_1$. We have $x_{K+1} = A^K x_1 + u_K$. Now for $t > K$ we also have $u_t = 0$, so $x_{t+1} = Ax_t$ and

$$x_t = A^{t-K-1}(A^K x_1 + u_K) = A^{t-1}x_1 + A^{t-K-1}u_K,$$

so $x_N = A^{N-1}x_1 + A^{N-K-1}u_K$. We solve $x_N = x^{\mathrm{des}}$ for u_K to get

$$u_K = A^{-N+K+1}(x^{\mathrm{des}} - A^{N-1}x_1).$$

The power of A is negative, but it exists since A is invertible.

11.20 *Immigration.* The population dynamics of a country is given by $x_{t+1} = Ax_t + u$, $t = 1, \ldots, T-1$, where the 100-vector x_t gives the population age distribution in year t, and u gives the immigration age distribution (with negative entries meaning emigration), which we assume is constant (*i.e.*, does not vary with t). You are given A, x_1, and x^{des}, a 100-vector that represents a desired population distribution in year T. We seek a constant level of immigration u that achieves $x_T = x^{\mathrm{des}}$.

Give a matrix formula for u. If your formula only makes sense when some conditions hold (for example invertibility of one or more matrices), say so.

Solution. We have

$$\begin{aligned}
x_2 &= Ax_1 + u \\
x_3 &= Ax_2 + u \\
&= A^2x_1 + Au + u \\
x_4 &= Ax_3 + u \\
&= A^3x_1 + A^2u + Au + u \\
&\vdots \\
x_T &= A^{T-1}x_1 + (A^{T-2} + \cdots + A + I)u.
\end{aligned}$$

so $x_4 = x^{\mathrm{des}}$ occurs when

$$u = (A^{T-2} + \cdots + A + I)^{-1}(x^{\mathrm{des}} - A^{T-1}x_1).$$

This requires that $A^{T-2} + \cdots + A + I$ be invertible.

11.21 *Quadrature weights.* Consider a quadrature problem (see exercise 8.12) with $n = 4$, with points $t = (-0.6, -0.2, 0.2, 0.6)$. We require that the quadrature rule be exact for all polynomials of degree up to $d = 3$.

Set this up as a square system of linear equations in the weight vector. Numerically solve this system to get the weights. Compute the true value and the quadrature estimate,

$$\alpha = \int_{-1}^{1} f(x)\,dx, \qquad \hat{\alpha} = w_1 f(-0.6) + w_2 f(-0.2) + w_3 f(0.2) + w_4 f(0.6),$$

for the specific function $f(x) = e^x$.

Solution. We solve the equation

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ t_1 & t_2 & t_3 & t_4 \\ t_1^2 & t_2^2 & t_3^2 & t_4^2 \\ t_1^3 & t_2^3 & t_3^3 & t_4^3 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}.$$

where

$$b_k = \int_{-1}^{1} t^{k-1} dt = \begin{cases} 2/k & k \text{ is odd} \\ 0 & k \text{ is even.} \end{cases}$$

The solution is

$$w_1 = 1.4375, \quad w_2 = -0.4375, \quad w_3 = -0.4375, \quad w_4 = 1.4375.$$

For $f(t) = e^x$ we find

$$\alpha = 2.3504, \qquad \hat{\alpha} = 2.5157.$$

11.22 *Properties of pseudo-inverses.* For an $m \times n$ matrix A and its pseudo-inverse $A^\dagger$, show that $A = AA^\dagger A$ and $A^\dagger = A^\dagger AA^\dagger$ in each of the following cases.

(a) A is tall with linearly independent columns.

(b) A is wide with linearly independent rows.

(c) A is square and invertible.

Solution.

(a) For a tall matrix A with linearly independent columns, $A^\dagger$ is a left inverse of A, so $A^\dagger A = I$. Therefore

$$AA^\dagger A = A(A^\dagger A) = AI = A, \qquad A^\dagger AA^\dagger = (A^\dagger A)A^\dagger = IA^\dagger = A^\dagger.$$

(b) For a wide matrix A with linearly independent rows, $A^\dagger$ is a right inverse of A, so $AA^\dagger = I$. Therefore

$$AA^\dagger A = (AA^\dagger)A = IA = A, \qquad A^\dagger AA^\dagger = A^\dagger(AA^\dagger) = A^\dagger I = A^\dagger.$$

(c) For a square invertible A, $A^\dagger = A^{-1}$, so

$$AA^\dagger A = AA^{-1}A = AI = A, \qquad A^\dagger AA^\dagger = A^{-1}AA^{-1} = IA^{-1} = A^{-1} = A^\dagger.$$

11.23 *Product of pseudo-inverses.* Suppose A and D are right-invertible matrices and the product AD exists. We have seen that if B is a right inverse of A and E is a right inverse of D, then EB is a right inverse of AD. Now suppose B is the pseudo-inverse of A and E is the pseudo-inverse of D. Is EB the pseudo-inverse of AD? Prove that this is always true or give an example for which it is false.

Solution. The result is not true in general. An example is

$$A = \begin{bmatrix} 1 & 1 \end{bmatrix}, \qquad D = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}, \qquad AD = \begin{bmatrix} 1 & 2 \end{bmatrix}.$$

The pseudo-inverses are

$$A^\dagger = \frac{1}{2}\begin{bmatrix} 1 \\ 1 \end{bmatrix}, \qquad D^\dagger = D^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & 1/2 \end{bmatrix}, \qquad (AD)^\dagger = \frac{1}{5}\begin{bmatrix} 1 \\ 2 \end{bmatrix}.$$

However

$$D^\dagger A^\dagger = \begin{bmatrix} 1/2 \\ 1/4 \end{bmatrix} \neq (AD)^\dagger.$$

11.24 *Simultaneous left inverse.* The two matrices

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 1 \\ 2 & 1 \\ 2 & 2 \end{bmatrix}, \qquad B = \begin{bmatrix} 3 & 2 \\ 1 & 0 \\ 2 & 1 \\ 1 & 3 \end{bmatrix}$$

and both left-invertible, and have multiple left inverses. Do they have a common left inverse? Explain how to find a 2×4 matrix C that satisfies $CA = CB = I$, or determine that no such matrix exists. (You can use numerical computing to find C.) *Hint.* Set up a set of linear equations for the entries of C. *Remark.* There is nothing special about the particular entries of the two matrices A and B.

Solution. C is a simultaneous left inverse if

$$C \begin{bmatrix} A & B \end{bmatrix} = \begin{bmatrix} I & I \end{bmatrix}.$$

We transpose the equation to put it in the usual form with the unknowns to the right of the coefficient matrix:

$$\begin{bmatrix} A^T \\ B^T \end{bmatrix} C^T = \begin{bmatrix} I \\ I \end{bmatrix},$$

or

$$\begin{bmatrix} 1 & 3 & 2 & 2 \\ 2 & 1 & 1 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 0 & 1 & 3 \end{bmatrix} \begin{bmatrix} C_{11} & C_{21} \\ C_{12} & C_{22} \\ C_{13} & C_{23} \\ C_{14} & C_{24} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

The solution is

$$C = \begin{bmatrix} -0.27273 & -0.18182 & 1.09091 & -0.18182 \\ 0.54545 & 0.36364 & -1.18182 & 0.36364 \end{bmatrix}.$$

11.25 *Checking the computed solution of linear equations.* One of your colleagues says that whenever you compute the solution x of a square set of n equations $Ax = b$ (say, using QR factorization), you should compute the number $\|Ax-b\|$ and check that it is small. (It is not exactly zero due to the small rounding errors made in floating point computations.) Another colleague says that this would be nice to do, but the additional cost of computing $\|Ax - b\|$ is too high. Briefly comment on your colleagues' advice. Who is right?

Solution. Computing the solution of $Ax = b$ costs around $2n^3$ flops. Computing $\|Ax-b\|$ costs around $2n^2$ flops. So checking the computed solution is around a factor n faster than computing the solution. For n bigger than around 30 or so, the additional cost of checking is not significant. So the first colleague is right, and the second one is wrong.

You might imagine that most people follow the first colleague's advice. In fact, almost no one does. But they should.

11.26 *Sensitivity of solution of linear equations.* Let A be an invertible $n \times n$ matrix, and b and x be n-vectors satisfying $Ax = b$. Suppose we perturb the jth entry of b by $\epsilon \neq 0$ (which is a traditional symbol for a small quantity), so b becomes $\tilde{b} = b + \epsilon e_j$. Let $\tilde{x}$ be the n-vector that satisfies $A\tilde{x} = \tilde{b}$, *i.e.*, the solution of the linear equations using the perturbed right-hand side. We are interested in $\|x - \tilde{x}\|$, which is how much the solution changes due to the change in the right-hand side. The ratio $\|x-\tilde{x}\|/|\epsilon|$ gives the sensitivity of the solution to changes (perturbations) of the jth entry of b.

(a) Show that $\|x - \tilde{x}\|$ does not depend on b; it only depends on the matrix A, ϵ, and j.

(b) How would you find the index j that maximizes the value of $\|x - \tilde{x}\|$? By part (a), your answer should be in terms of A (or quantities derived from A) and ϵ only.

Remark. If a small change in the right-hand side vector b can lead to a large change in the solution, we say that the linear equations $Ax = b$ are *poorly conditioned* or *ill-conditioned*. As a practical matter it means that unless you are very confident in what the entries of b are, the solution $A^{-1}b$ may not be useful in practice.

Solution.

(a) The solutions of the original and the perturbed equations are

$$x = A^{-1}b, \qquad \tilde{x} = A^{-1}(b + \epsilon e_j) = A^{-1}b + \epsilon A^{-1}e_j.$$

Note that $A^{-1}e_j$ is the jth column of A^{-1}. The distance between the two solutions is

$$\|x - \tilde{x}\| = \|\epsilon A^{-1}e_j\| = |\epsilon|\|A^{-1}e_j\|.$$

(b) We find the column of A^{-1} with the largest norm.

11.27 *Timing test.* Generate a random $n \times n$ matrix A and an n-vector b, for $n = 500$, $n = 1000$, and $n = 2000$. For each of these, compute the solution $x = A^{-1}b$ (for example using the backslash operator, if the software you are using supports it), and verify that $Ax - b$ is (very) small. Report the time it takes to solve each of these three sets of linear equations, and for each one work out the implied speed of your processor in Gflop/s, based on the $2n^3$ complexity of solving equations using the QR factorization.

Solution. Here is a simple script in Julia.

```
for n in [500,1000,2000]
A = randn(n,n);
b = randn(n);
@time x=A\b;
end
```

On Boyd's laptop, this produces the output

```
0.007727 seconds (8 allocations: 1.915 MiB)
0.029618 seconds (8 allocations: 7.645 MiB)
0.156405 seconds (8 allocations: 30.549 MiB, 2.76% gc time)
```

Of course, your numbers could vary from these. Using $2n^3$ as the complexity, the numbers of Gflops in these three computations are 0.25, 2, and 16, respectively. The implied flop rates are around 32, 68, and 100, in Gflop/sec. It's not unusual to have variations like these in implied flop counts; our simple complexity calculations are only meant to estimate the execution time roughly.

11.28 *Solving multiple linear equations efficiently.* Suppose the $n \times n$ matrix A is invertible. We can solve the system of linear equations $Ax = b$ in around $2n^3$ flops using algorithm 11.2. Once we have done that (specifically, computed the QR factorization of A), we can solve an additional set of linear equations with same matrix but different right-hand side, $Ay = c$, in around $3n^2$ additional flops. Assuming we have solved both of these sets of equations, suppose we want to solve $Az = d$, where $d = \alpha b + \beta c$ is a linear combination of b and c. (We are given the coefficients α and β.) Suggest a method for doing this that is even faster than re-using the QR factorization of A. Your method should have a complexity that is *linear* in n. Give rough estimates for the time needed to solve $Ax = b$, $Ay = c$, and $Az = d$ (using your method) for $n = 3000$ on a computer capable of carrying out 1 Gflop/s.

Solution. We observe that

$$z = A^{-1}(\alpha b + \beta c) = \alpha(A^{-1}b) + \beta(A^{-1}c) = \alpha x + \beta y.$$

We can simply form this linear combination of the solutions already computed. The cost is $3n$ flops.

The costs of computing x, y, and z are around $2n^3$, $3n^2$, and $3n$ flops, respectively. Associated times with a 1 Gflop/s computer are around one minute, 27 milliseconds, and 9 microseconds, respectively.

Chapter 12

Least squares

Exercises

12.1 *Approximating a vector as a multiple of another one.* In the special case $n = 1$, the general least squares problem (12.1) reduces to finding a scalar x that minimizes $\|ax-b\|^2$, where a and b are m-vectors. (We write the matrix A here in lower case, since it is an m-vector.) Assuming a and b are nonzero, show that $\|a\hat{x} - b\|^2 = \|b\|^2(\sin\theta)^2$, where $\theta = \angle(a, b)$. This shows that the optimal relative error in approximating one vector by a multiple of another one depends on their angle.

Solution. Using the general formula we have

$$\hat{x} = (a^T a)^{-1} a^T b = \frac{a^T b}{\|a\|^2},$$

so the optimal error is given by

$$\begin{aligned}
\|a\hat{x} - b\|^2 &= \left\| a\frac{a^T b}{\|a\|^2} - b \right\|^2 \\
&= \|a\|^2 \frac{(a^T b)^2}{\|a\|^4} - 2\frac{(a^T b)^2}{\|a\|^2} + \|b\|^2 \\
&= \|b\|^2 \left(1 - \frac{a^T b}{\|a\|^2\|b\|^2}\right) \\
&= \|b\|^2 \left(1 - (\cos\theta)^2\right) \\
&= \|b\|^2 (\sin\theta)^2.
\end{aligned}$$

12.2 *Least squares with orthonormal columns.* Suppose the $m \times n$ matrix Q has orthonormal columns and b is an m-vector. Show that $\hat{x} = Q^T b$ is the vector that minimizes $\|Qx-b\|^2$. What is the complexity of computing $\hat{x}$, given Q and b, and how does it compare to the complexity of a general least squares problem with an $m \times n$ coefficient matrix?

Solution. The columns of Q are linearly independent, so the general formula gives

$$\hat{x} = (Q^T Q)^{-1} Q^T b = IQ^T b = Q^T b,$$

using $Q^T Q = I$.

To compute $\hat{x}$, we simply multiply b by Q^T, which costs $2mn$ flops. In the general case, solving a least squares problem with the same dimensions costs $2mn^2$ flops, which is a factor of n more than the special case when the columns of the coefficient matrix are orthonormal.

12.3 *Least angle property of least squares.* Suppose the $m \times n$ matrix A has linearly independent columns, and b is an m-vector. Let $\hat{x} = A^\dagger b$ denote the least squares approximate solution of $Ax = b$.

(a) Show that for any n-vector x, $(Ax)^T b = (Ax)^T (A\hat{x})$, *i.e.*, the inner product of Ax and b is the same as the inner product of Ax and $A\hat{x}$. *Hint.* Use $(Ax)^T b = x^T (A^T b)$ and $(A^T A)\hat{x} = A^T b$.

(b) Show that when $A\hat{x}$ and b are both nonzero, we have

$$\frac{(A\hat{x})^T b}{\|A\hat{x}\|\|b\|} = \frac{\|A\hat{x}\|}{\|b\|}.$$

The left-hand side is the cosine of the angle between $A\hat{x}$ and b. *Hint.* Apply part (a) with $x = \hat{x}$.

(c) *Least angle property of least squares.* The choice $x = \hat{x}$ minimizes the distance between Ax and b. Show that $x = \hat{x}$ also minimizes the angle between Ax and b. (You can assume that Ax and b are nonzero.) *Remark.* For any positive scalar α, $x = \alpha\hat{x}$ also minimizes the angle between Ax and b.

Solution.

(a) To see this, we note that

$$(Ax)^T b = x^T(A^T b) = x^T(A^T A\hat{x}) = (Ax)^T(A\hat{x}),$$

using the normal equations $(A^T A)\hat{x} = A^T b$.

(b) We apply part (a) with $x = \hat{x}$ to get $(A\hat{x})^T b = \|A\hat{x}\|^2$. Divide this by $\|A\hat{x}\|\|b\|$ to get the equality.

(c) We have to show that for any x, we have

$$\cos^{-1}\left(\frac{(A\hat{x})^T b}{\|A\hat{x}\|\|b\|}\right) \leq \cos^{-1}\left(\frac{(Ax)^T b}{\|Ax\|\|b\|}\right),$$

provided $Ax \neq 0$. Using part (a) this is the same as

$$\frac{(A\hat{x})^T(A\hat{x})}{\|A\hat{x}\|\|b\|} \geq \frac{(Ax)^T(A\hat{x})}{\|Ax\|\|b\|}.$$

We can cancel the term $\|b\|$ from each side, and re-arrange this as

$$\|A\hat{x}\|\|Ax\| \geq (Ax)^T(A\hat{x}),$$

which is true by the Cauchy–Schwarz inequality.

12.4 *Weighted least squares.* In least squares, the objective (to be minimized) is

$$\|Ax - b\|^2 = \sum_{i=1}^{m}(\tilde{a}_i^T x - b_i)^2,$$

where $\tilde{a}_i^T$ are the rows of A, and the n-vector x is to chosen. In the *weighted least squares problem*, we minimize the objective

$$\sum_{i=1}^{m} w_i(\tilde{a}_i^T x - b_i)^2,$$

where w_i are given positive weights. The weights allow us to assign different weights to the different components of the residual vector. (The objective of the weighted least squares problem is the square of the weighted norm, $\|Ax - b\|_w^2$, as defined in exercise 3.28.)

(a) Show that the weighted least squares objective can be expressed as $\|D(Ax - b)\|^2$ for an appropriate diagonal matrix D. This allows us to solve the weighted least squares problem as a standard least squares problem, by minimizing $\|Bx - d\|^2$, where $B = DA$ and $d = Db$.

(b) Show that when A has linearly independent columns, so does the matrix B.

(c) The least squares approximate solution is given by $\hat{x} = (A^T A)^{-1}A^T b$. Give a similar formula for the solution of the weighted least squares problem. You might want to use the matrix $W = \mathbf{diag}(w)$ in your formula.

Solution.

(a) Since the weights are positive, we can write the objective as

$$\sum_{i=1}^{m} w_i(\tilde{a}_i^T x - b_i)^2 = \sum_{i=1}^{m} (\sqrt{w_i}(\tilde{a}_i^T x - b_i))^2 = \|D(Ax-b)\|^2,$$

where D is the diagonal matrix

$$D = \begin{bmatrix} \sqrt{w_1} & 0 & \cdots & 0 \\ 0 & \sqrt{w_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sqrt{w_m} \end{bmatrix}.$$

(b) We show that $Bx = 0$ implies $x = 0$.
Suppose $Bx = DAx = 0$. Then $Ax = 0$ because D is a diagonal matrix with positive diagonal entries, and hence invertible. By assumption, A has linearly independent columns, so $Ax = 0$ implies $x = 0$.

(c) The solution of the weighted least squares problem is

$$\begin{aligned} (B^TB)^{-1}B^Td &= ((DA)^T(DA))^{-1}(DA)^T(Db) \\ &= (A^TD^2A)^{-1}(A^TD^2b) \\ &= (A^TWA)^{-1}(A^TWb), \end{aligned}$$

where $W = D^2 = \mathbf{diag}(w)$.

12.5 *Approximate right inverse.* Suppose the tall $m \times n$ matrix A has linearly independent columns. It does not have a right inverse, *i.e.*, there is no $n \times m$ matrix X for which $AX = I$. So instead we seek the $n \times m$ matrix X for which the residual matrix $R = AX - I$ has the smallest possible matrix norm. We call this matrix the *least squares approximate right inverse* of A. Show that the least squares right inverse of A is given by $X = A^\dagger$. *Hint.* This is a matrix least squares problem; see page 233.

Solution. The jth column of the $m \times m$ matrix R is $Ax_j - e_j$ where x_j is the jth column of A and e_j is the jth unit vector. The square of the matrix norm of R is the sum of the squared column norms:

$$\|R\|^2 = \sum_{j=1}^{m} \|r_j\|^2 = \sum_{j=1}^{m} \|Ax_j - e_j\|^2.$$

Each term in the sum depends on one column of X and can be minimized independently of the other columns. Therefore we choose x_j to minimize $\|Ax_j - e_j\|^2$, which is a least squares problem with solution

$$\hat{x}_j = A^\dagger e_j,$$

i.e., the jth column of $A^\dagger$. The approximate inverse $\hat{X}$ is the matrix

$$\begin{aligned} \hat{X} &= \begin{bmatrix} \hat{x}_1 & \hat{x}_2 & \cdots & \hat{x}_m \end{bmatrix} \\ &= \begin{bmatrix} A^\dagger e_1 & A^\dagger e_2 & \cdots & A^\dagger e_m \end{bmatrix} \\ &= A^\dagger \begin{bmatrix} e_1 & e_2 & \cdots & e_m \end{bmatrix} \\ &= A^\dagger I \\ &= A^\dagger. \end{aligned}$$

12.6 *Least squares equalizer design.* (See exercise 7.15.) You are given a channel impulse response, the n-vector c. Your job is to find an equalizer impulse response, the n-vector h, that minimizes $\|h * c - e_1\|^2$. You can assume that $c_1 \neq 0$. *Remark.* h is called an equalizer since it approximately inverts, or undoes, convolution by c.

Explain how to find h. Apply your method to find the equalizer h for the channel $c = (1.0, 0.7, -0.3, -0.1, 0.05)$. Plot c, h, and $h * c$.

Solution.

(a) Recall that the convolution $c * h = h * c$ is a linear function of h and can be written as a matrix-vector product

$$c * h = T(c)h,$$

where $T(c)$ is the Toeplitz matrix (7.3). Here, $m = n$ so the matrix has size $(2n - 1) \times n$, with

$$T(c) = \begin{bmatrix} c_1 & 0 & \cdots & 0 \\ c_2 & c_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ c_n & c_{n-1} & \cdots & c_1 \\ 0 & c_n & \cdots & c_2 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & c_n. \end{bmatrix}$$

Therefore $\|c*h-e_1\|^2 = \|T(c)h-e_1\|^2$ and we can find h by solving the least squares problem of minimizing

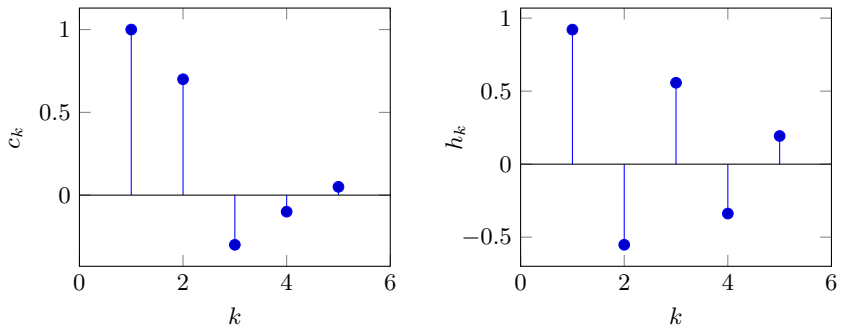
$$\|T(c)h - e_1\|^2.$$

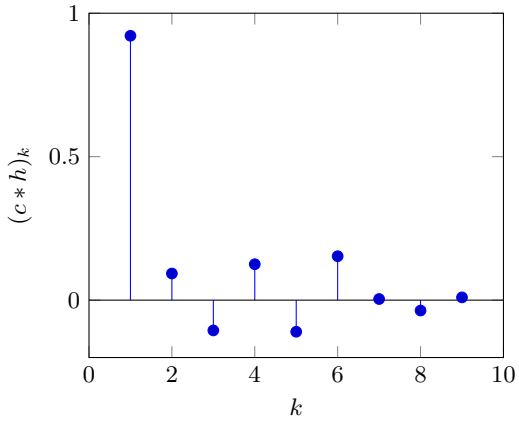
(b) The solution is

$$h = (0.9214,\ -0.55218,\ 0.55739,\ -0.33847,\ 0.19274).$$

The convolution with the channel impulse response is

$$h*c = (0.9214,\ 0.0928,\ -0.1056,\ 0.1252,\ -0.1101,\ 0.1531,\ 0.0039,\ -0.0362,\ 0.0096).$$





12.7 *Network tomography.* A network consists of n links, labeled $1, \ldots, n$. A *path* through the network is a subset of the links. (The order of the links on a path does not matter here.) Each link has a (positive) *delay*, which is the time it takes to traverse it. We let d denote the n-vector that gives the link delays. The total travel time of a path is the sum of the delays of the links on the path. Our goal is to estimate the link delays (*i.e.*, the vector d), from a large number of (noisy) measurements of the travel times along different paths. This data is given to you as an $N \times n$ matrix P, where

$$P_{ij} = \begin{cases} 1 & \text{link } j \text{ is on path } i \\ 0 & \text{otherwise,} \end{cases}$$

and an N-vector t whose entries are the (noisy) travel times along the N paths. You can assume that $N > n$. You will choose your estimate $\hat{d}$ by minimizing the RMS deviation between the measured travel times (t) and the travel times predicted by the sum of the link delays. Explain how to do this, and give a matrix expression for $\hat{d}$. If your expression requires assumptions about the data P or t, state them explicitly.

Remark. This problem arises in several contexts. The network could be a computer network, and a path gives the sequence of communication links data packets traverse. The network could be a transportation system, with the links representing road segments.

Solution. For given link delays d, the travel times predicted by the sum of the link delays is Pd. The RMS deviation between this prediction and the measured travel times is $\|Pd-t\|/\sqrt{N}$. Minimizing this expression is equivalent to minimizing $\|Pd-t\|^2$, which is a least squares problem. The estimate is therefore

$$\hat{d} = (P^TP)^{-1}P^Tt = P^\dagger t.$$

This assumes that the matrix P has linearly independent columns.

12.8 *Least squares and QR factorization.* Suppose A is an $m \times n$ matrix with linearly independent columns and QR factorization $A = QR$, and b is an m-vector. The vector $A\hat{x}$ is the linear combination of the columns of A that is closest to the vector b, *i.e.*, it is the projection of b onto the set of linear combinations of the columns of A.

(a) Show that $A\hat{x} = QQ^Tb$. (The matrix QQ^T is called the *projection matrix*.)

(b) Show that $\|A\hat{x}-b\|^2 = \|b\|^2 - \|Q^Tb\|^2$. (This is the square of the distance between b and the closest linear combination of the columns of A.)

Solution.

(a) Using the expression $\hat{x} = A^\dagger b = R^{-1}Q^Tb$, we find

$$A\hat{x} = AA^\dagger b = QRR^{-1}Q^Tb = QQ^Tb.$$

(b) We first use the normal equations $A^TA\hat{x} = A^Tb$ to find an interesting expression for the square norm of the least squares residual:

$$\begin{aligned}\|A\hat{x} - b\|^2 &= \|b\|^2 + \|A\hat{x}\|^2 - \hat{x}^TA^Tb \\ &= \|b\|^2 + \|A\hat{x}\|^2 - \hat{x}^TA^TA\hat{x} \\ &= \|b\|^2 + \|A\hat{x}\|^2 - 2\|A\hat{x}\|^2 \\ &= \|b\|^2 - \|A\hat{x}\|^2.\end{aligned}$$

Now we substitute the expression for $A\hat{x}$ derived in part (a):

$$\|A\hat{x} - b\|^2 = \|b\|^2 - \|QQ^Tb\|^2 = \|b\|^2 - \|Q^Tb\|^2.$$

The last simplification follows because Q has orthonormal columns, and therefore $\|Qy\|^2 = y^TQ^TQy = y^Ty = \|y\|^2$ for all y.

12.9 *Invertibility of matrix in sparse least squares formulation.* Show that the $(m+n)\times(m+n)$ coefficient matrix appearing in equation (12.11) is invertible if and only if the columns of A are linearly independent.

Solution. Suppose the columns of A are linearly independent. Assume that x, y satisfy

$$\begin{bmatrix} 0 & A^T \\ A & I \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Then $A^Ty = 0$ and $Ax + y = 0$. Substituting $y = -Ax$ in the first equation gives $A^TAx = 0$. Therefore $x^TA^TAx = \|Ax\|^2 = 0$, *i.e.*, $Ax = 0$. Since the columns of A are linearly independent, this is only possible if $x = 0$. If $x = 0$, then also $y = -Ax = 0$. We have shown that

$$\begin{bmatrix} 0 & A^T \\ A & I \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

only holds if $x = 0$ and $y = 0$. This means the matrix is invertible.

Next, suppose the columns of A are linearly dependent. Then there exists a nonzero x with $Ax = 0$. We have

$$\begin{bmatrix} 0 & A^T \\ A & I \end{bmatrix}\begin{bmatrix} x \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

and since $(x, 0) \neq 0$ this shows the matrix is not invertible.

12.10 *Numerical check of the least squares approximate solution.* Generate a random 30×10 matrix A and a random 30-vector b. Compute the least squares approximate solution $\hat{x} = A^\dagger b$ and the associated residual norm squared $\|A\hat{x} - b\|^2$. (There may be several ways to do this, depending on the software package you use.) Generate three different random 10-vectors d_1, d_2, d_3, and verify that $\|A(\hat{x} + d_i) - b\|^2 > \|A\hat{x} - b\|^2$ holds. (This shows that $x = \hat{x}$ has a smaller associated residual than the choices $x = \hat{x}+d_i$, $i = 1, 2, 3$.)

12.11 *Complexity of matrix least squares problem.* Explain how to compute the matrix least squares approximate solution of $AX = B$, given by $\hat{X} = A^\dagger B$ (see (12.12)), in no more than $2mn^2 + 3mnk$ flops. (In contrast, solving k vector least squares problems to obtain the columns of $\hat{X}$, in a naïve way, requires $2mn^2k$ flops.)

Solution. We compute the QR factorization of A once, which costs $2mn^2$ flops. Then we compute the columns of $\hat{X}$ by back substitution, solving $R\hat{x}_j = (Q^TB)_j$, where the right-hand side is the jth column of Q^TB. Forming Q^TB costs $2mnk$ flops, and the k back solves cost kn^2 flops. Adding it all up we get $2mn^2+2mnk+kn^2$ flops. Since $m \geq n$, the last term is no more than kmn, so the total is no more than $2mn^2 + 3mnk$ flops.

12.12 *Least squares placement.* The 2-vectors $p_1, \ldots, p_N$ represent the locations or positions of N objects, for example, factories, warehouses, and stores. The last K of these locations are fixed and given; the goal in a *placement problem* to choose the locations of the first $N - K$ objects. Our choice of the locations is guided by an undirected graph; an edge between two objects means we would like them to be close to each other. In *least squares placement*, we choose the locations $p_1, \ldots, p_{N-K}$ so as to minimize the sum of the squares of the distances between objects connected by an edge,

$$\|p_{i_1} - p_{j_1}\|^2 + \cdots + \|p_{i_L} - p_{j_L}\|^2,$$

where the L edges of the graph are given by $(i_1, j_1), \ldots, (i_L, j_L)$.

(a) Let $\mathcal{D}$ be the Dirichlet energy of the graph, as defined on page 135. Show that the sum of the squared distances between the N objects can be expressed as $\mathcal{D}(u)+\mathcal{D}(v)$, where $u = ((p_1)_1, \ldots, (p_N)_1)$ and $v = ((p_1)_2, \ldots, (p_N)_2)$ are N-vectors containing the first and second coordinates of the objects, respectively.

(b) Express the least squares placement problem as a least squares problem, with variable $x = (u_{1:(N-K)}, v_{1:(N-K)})$. In other words, express the objective above (the sum of squares of the distances across edges) as $\|Ax-b\|^2$, for an appropriate $m \times n$ matrix A and m-vector b. You will find that $m = 2L$. *Hint.* Recall that $\mathcal{D}(y) = \|B^T y\|^2$, where B is the incidence matrix of the graph.

(c) Solve the least squares placement problem for the specific problem with $N = 10$, $K = 4$, $L = 13$, fixed locations

$$p_7 = (0, 0), \quad p_8 = (0, 1), \quad p_8 = (1, 1), \quad p_{10} = (1, 0),$$

and edges

$$(1,3), \quad (1,4), \quad (1,7), \quad (2,3), \quad (2,5), \quad (2,8), \quad (2,9),$$
$$(3,4), \quad (3,5), \quad (4,6), \quad (5,6), \quad (6,9), \quad (6,10).$$

Plot the locations, showing the graph edges as lines connecting the locations.

Solution.

(a) The sum of the squared distances is

$$\begin{aligned} &\|p_{i_1} - p_{j_1}\|^2 + \cdots + \|p_{i_L} - p_{j_L}\|^2 \\ &= \left\| \begin{bmatrix} u_{i_1} - u_{j_1} \\ v_{i_1} - v_{j_1} \end{bmatrix} \right\|^2 + \cdots + \left\| \begin{bmatrix} u_{i_L} - u_{j_L} \\ v_{i_L} - v_{j_L} \end{bmatrix} \right\|^2 \\ &= (u_{i_1} - u_{j_1})^2 + \cdots + (u_{i_L} - u_{j_L})^2 + (v_{i_1} - v_{j_1})^2 + \cdots + (v_{i_L} - v_{j_L})^2 \\ &= \mathcal{D}(u) + \mathcal{D}(v). \end{aligned}$$

(b) We have $\mathcal{D}(u) + \mathcal{D}(v) = \|B^T u\|^2 + \|B^T v\|^2$ where B is the $N \times L$ node incidence matrix

$$B_{ij} = \begin{cases} 1 & \text{edge } j \text{ points to node } i \\ -1 & \text{edge } j \text{ points from node } i \\ 0 & \text{otherwise.} \end{cases}$$

Suppose we partition B as

$$B = \begin{bmatrix} B_{\mathrm{m}} \\ B_{\mathrm{f}} \end{bmatrix}, \qquad B_{\mathrm{m}} = B_{1:(N-K),1:L}, \qquad B_{\mathrm{f}} = B_{(N-K+1):N,1:L},$$

and the vectors u and v as $u = (u_{\mathrm{m}}, u_{\mathrm{f}})$ and $v = (v_{\mathrm{m}}, v_{\mathrm{f}})$ where

$$u_{\mathrm{m}} = u_{1:(N-K)}, \qquad v_{\mathrm{m}} = v_{1:(N-K)}, \qquad u_{\mathrm{f}} = u_{(N-K+1):N}, \qquad v_{\mathrm{f}} = v_{(N-K+1):N}.$$

Then the objective can be expressed as

$$\begin{aligned}\left\|B^T u\right\|^2 + \left\|B^T v\right\|^2 &= \left\|B_{\mathrm{m}}^T u_{\mathrm{m}} + B_{\mathrm{f}}^T u_{\mathrm{f}}\right\|^2 + \left\|B_{\mathrm{m}}^T v_{\mathrm{m}} + B_{\mathrm{f}}^T v_{\mathrm{f}}\right\|^2 \\ &= \left\| \begin{bmatrix} B_{\mathrm{m}}^T & 0 \\ 0 & B_{\mathrm{m}}^T \end{bmatrix} \begin{bmatrix} u_{\mathrm{m}} \\ v_{\mathrm{m}} \end{bmatrix} + \begin{bmatrix} B_{\mathrm{f}}^T u_{\mathrm{f}} \\ B_{\mathrm{f}}^T v_{\mathrm{f}} \end{bmatrix} \right\|^2 .\end{aligned}$$

Minimizing this is a least squares problem with variable $x = (u_{\mathrm{m}}, v_{\mathrm{m}})$,

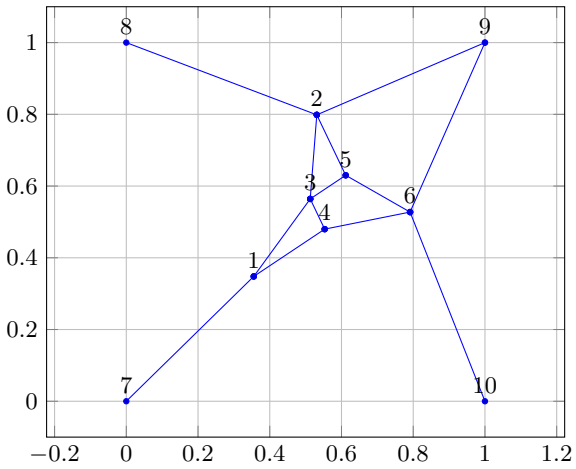
$$A = \begin{bmatrix} B_{\mathrm{m}}^T & 0 \\ 0 & B_{\mathrm{m}}^T \end{bmatrix}, \qquad b = -\begin{bmatrix} B_{\mathrm{f}}^T u_{\mathrm{f}} \\ B_{\mathrm{f}}^T v_{\mathrm{f}} \end{bmatrix}.$$

Note that the problem is also equivalent to two independent least squares problems

$$\text{minimize } \|B_{\mathrm{m}}^T u_{\mathrm{m}} + B_{\mathrm{f}}^T u_{\mathrm{f}}\|^2, \qquad \text{minimize } \|B_{\mathrm{m}}^T v_{\mathrm{m}} + B_{\mathrm{f}}^T v_{\mathrm{f}}\|^2,$$

with variables u_{m} and v_{m}, respectively, and that the two problems have the same coefficient matrix B_{m}^T.

(c) The solution is shown in the figure.



12.13 *Iterative method for least squares problem.* Suppose that A has linearly independent columns, so $\hat{x} = A^\dagger b$ minimizes $\|Ax - b\|^2$. In this exercise we explore an iterative method, due to the mathematician Lewis Richardson, that can be used to compute $\hat{x}$. We define $x^{(1)} = 0$ and for $k = 1, 2, \ldots$,

$$x^{(k+1)} = x^{(k)} - \mu A^T (Ax^{(k)} - b),$$

where μ is a positive parameter, and the superscripts denote the iteration number. This defines a sequence of vectors that converge to $\hat{x}$ provided μ is not too large; the choice $\mu = 1/\|A\|^2$, for example, always works. The iteration is terminated when $A^T(Ax^{(k)} - b)$ is small enough, which means the least squares optimality conditions are almost satisfied. To implement the method we only need to multiply vectors by A and by A^T. If we have efficient methods for carrying out these two matrix-vector multiplications, this iterative method can be faster than algorithm 12.1 (although it does not give the exact solution). Iterative methods are often used for very large scale least squares problems.

(a) Show that if $x^{(k+1)} = x^{(k)}$, we have $x^{(k)} = \hat{x}$.

(b) Express the vector sequence $x^{(k)}$ as a linear dynamical system with constant dynamics matrix and offset, *i.e.*, in the form $x^{(k+1)} = Fx^{(k)} + g$.

(c) Generate a random 20×10 matrix A and 20-vector b, and compute $\hat{x} = A^\dagger b$. Run the Richardson algorithm with $\mu = 1/\|A\|^2$ for 500 iterations, and plot $\|x^{(k)} - \hat{x}\|$ to verify that $x^{(k)}$ appears to be converging to $\hat{x}$.

Solution.

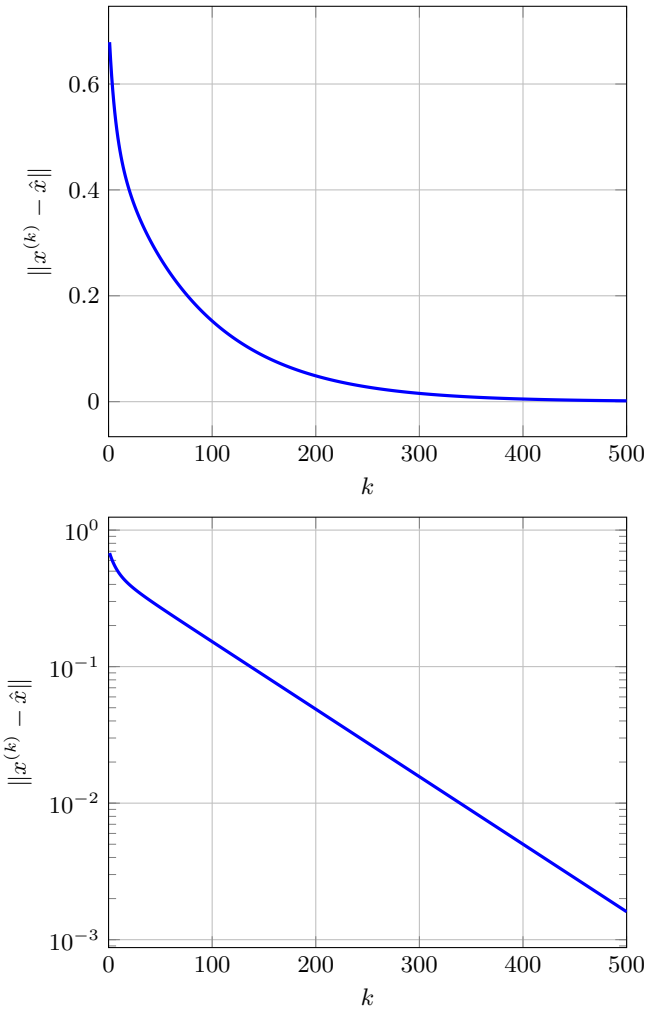
(a) If $x^{(k+1)} = x^{(k)}$, then $A^T(Ax^{(k)} - b) = 0$, so $x^{(k)} = \hat{x}$. (This is because $\hat{x}$ is the unique solution of the normal equations $A^T(Ax - b) = 0$.)

(b) We have

$$\begin{aligned} x^{(k+1)} &= x^{(k)} - \mu A^T(Ax^{(k)} - b) \\ &= (I - \mu A^T A)x^{(k)} + \mu A^T b \\ &= Fx^{(k)} + g \end{aligned}$$

with $F = I - \mu A^T A$, $g = A^T b$.

(c) The plots show the error $\|x^{(k)} - \hat{x}\|$ on a linear and a logarithmic scale.



12.14 *Recursive least squares.* In some applications of least squares the rows of the coefficient matrix A become available (or are added) sequentially, and we wish to solve the resulting family of growing least squares problems. Define the $k \times n$ matrices and k-vectors

$$A^{(k)} = \begin{bmatrix} a_1^T \\ \vdots \\ a_k^T \end{bmatrix} = A_{1:k,1:n}, \qquad b^{(k)} = \begin{bmatrix} b_1 \\ \vdots \\ b_k \end{bmatrix} = b_{1:k},$$

for $k = 1, \ldots, m$. We wish to compute $\hat{x}^{(k)} = A^{(k)\dagger} b^{(k)}$, for $k = n, n+1, \ldots, m$. We will assume that the columns of $A^{(n)}$ are linearly independent, which implies that the columns of $A^{(k)}$ are linearly independent for $k = n, \ldots, m$. We will also assume that m is much larger than n. The naïve method for computing $x^{(k)}$ requires $2kn^2$ flops, so the total cost for $k = n, \ldots, m$ is

$$\sum_{k=n}^{m} 2kn^2 = \left(\sum_{k=n}^{m} k \right) (2n^2) = \left(\frac{m^2 - n^2 + m + n}{2} \right) (2n^2) \approx m^2 n^2 \text{ flops.}$$

A simple trick allows us to compute $x^{(k)}$ for $k = n \ldots, m$ much more efficiently, with a cost that grows linearly with m. The trick also requires memory storage order n^2, which does not depend on m. for $k = 1, \ldots, m$, define

$$G^{(k)} = (A^{(k)})^T A^{(k)}, \qquad h^{(k)} = (A^{(k)})^T b^{(k)}.$$

(a) Show that $\hat{x}^{(k)} = (G^{(k)})^{-1} h^{(k)}$ for $k = n, \ldots, m$. *Hint.* See (12.8).

(b) Show that $G^{(k+1)} = G^{(k)} + a_k a_k^T$ and $h^{(k+1)} = h^{(k)} + b_k a_k$, for $k = 1, \ldots, m-1$.

(c) *Recursive least squares* is the following algorithm. For $k = n, \ldots, m$, compute $G^{(k+1)}$ and $h^{(k+1)}$ using (b); then compute $\hat{x}^{(k)}$ using (a). Work out the total flop count for this method, keeping only dominant terms. (You can include the cost of computing $G^{(n)}$ and $h^{(n)}$, which should be negligible in the total.) Compare to the flop count for the naïve method.

Remark. A further trick called the matrix inversion lemma (which is beyond the scope of this book) can be used to reduce the complexity of recursive least squares to order mn^2.

Solution.

(a) We have

$$\begin{aligned} \hat{x}^{(k)} &= A^{(k)\dagger} b^{(k)} \\ &= \left((A^{(k)})^T A^{(k)} \right)^{-1} (A^{(k)})^T b^{(k)} \\ &= (G^{(k)})^{-1} h^{(k)}. \end{aligned}$$

(b) We will show that $G^{(k)} = \sum_{i=1}^{k} a_i a_i^T$. Using block matrix transposition and multiplication,

$$G^{(k)} = (A^{(k)})^T A^{(k)} = \begin{bmatrix} a_1^T \\ \vdots \\ a_k^T \end{bmatrix}^T \begin{bmatrix} a_1^T \\ \vdots \\ a_k^T \end{bmatrix} = \begin{bmatrix} a_1 & \cdots & a_k \end{bmatrix} \begin{bmatrix} a_1^T \\ \vdots \\ a_k^T \end{bmatrix} = \sum_{i=1}^{k} a_i a_i^T.$$

(c) Computing $G^{(k)}$ costs $2n^2$ flops, and computing $h^{(k)}$ costs $2n$ flops (and so can be dropped). For $k = 1, \ldots, m$ this requires a total of $2mn^2$ flops. Computing each $\hat{x}^{(k)}$ costs $2n^3$ flops. For $k = n, \ldots, m$ the total is then $(m - n + 1)2n^2$ flops, which can be simplified to $2mn^3$ flops dropping smaller terms. So the total cost of carrying out recursive least squares is $2mn^3$ flops, which is linear in m, whereas the naïve method is quadratic in m.

12.15 *Minimizing a squared norm plus an affine function.* A generalization of the least squares problem (12.1) adds an affine function to the least squares objective,

$$\text{minimize} \quad \|Ax - b\|^2 + c^T x + d,$$

where the n-vector x is the variable to be chosen, and the (given) data are the $m \times n$ matrix A, the m-vector b, the n-vector c, and the number d. We will use the same assumption we use in least squares: The columns of A are linearly independent. This generalized problem can be solved by reducing it to a standard least squares problem, using a trick called *completing the square*.

Show that the objective of the problem above can be expressed in the form

$$\|Ax - b\|^2 + c^T x + d = \|Ax - b + f\|^2 + g,$$

for some m-vector f and some constant g. It follows that we can solve the generalized least squares problem by minimizing $\|Ax - (b - f)\|$, an ordinary least squares problem with solution $\hat{x} = A^\dagger(b - f)$.

Hints. Express the norm squared term on the right-hand side as $\|(Ax - b) + f\|^2$ and expand it. Then argue that the equality above holds provided $2A^T f = c$. One possible choice is $f = (1/2)(A^\dagger)^T c$. (You must justify these statements.)

Solution. Following the first hint we get

$$\|Ax - b\|^2 + c^T x + d = \|Ax - b\|^2 + 2f^T(Ax - b) + \|f\|^2 + g.$$

The first term on each side is the same. We write the remainder as

$$c^T x + d = 2(A^T f)^T x - 2f^T b + \|f\|^2 + g.$$

If $2A^T f = c$, the two linear terms are equal. We get equality when we choose

$$g = d + 2f^T b - \|f\|^2.$$

Finally, we need to show that the choice $f = (1/2)(A^\dagger)^T c$ satisfies $2A^T f = c$. With this choice of f,

$$2^T f = A^T((A^T A)^{-1} A^T)^T c = A^T A((A^T A)^{-1})^T c = A^T A(A^T A)^{-1} c = c.$$

12.16 *Gram method for computing least squares approximate solution.* Algorithm 12.1 uses the QR factorization to compute the least squares approximate solution $\hat{x} = A^\dagger b$, where the $m \times n$ matrix A has linearly independent columns. It has a complexity of $2mn^2$ flops. In this exercise we consider an alternative method: First, form the Gram matrix $G = A^T A$ and the vector $h = A^T b$; and then compute $\hat{x} = G^{-1}h$ (using algorithm 11.2). What is the complexity of this method? Compare it to algorithm 12.1. *Remark.* You might find that the Gram algorithm appears to be a bit faster than the QR method, but the factor is not large enough to have any practical significance. The idea is useful in situations where G is partially available and can be computed more efficiently than by multiplying A and its transpose. An example is exercise 13.21.

Solution. The multiplication $A^T A$ costs mn^2 flops. (Since the result is symmetric, we only need to calculate $n(n+1)/2$ entries. Each entry is an inner product of length m, so the total is $(2m-1)n(n+1)/2 \approx mn^2$.) The matrix-vector multiplication $A^T b$ costs $2mn$ flops. Solving $G\hat{x} = h$ costs $2n^3$ if we use the QR factorization of G. This gives a total of $mn^2 + 2n^3$. If m is much greater than n this is a factor of two less than the QR method.

Chapter 13

Least squares data fitting

Exercises

13.1 *Error in straight-line fit.* Consider the straight-line fit described on page 249, with data given by the N-vectors x^{d} and y^{d}. Let $r^{\mathrm{d}} = y^{\mathrm{d}} - \hat{y}^{\mathrm{d}}$ denote the residual or prediction error using the straight-line model (13.3). Show that $\mathbf{rms}(r^{\mathrm{d}}) = \mathbf{std}(y^{\mathrm{d}})\sqrt{1-\rho^2}$, where ρ is the correlation coefficient of x^{d} and y^{d} (assumed non-constant). This shows that the RMS error with the straight-line fit is a factor $\sqrt{1-\rho^2}$ smaller than the RMS error with a constant fit, which is $\mathbf{std}(y^{\mathrm{d}})$. It follows that when x^{d} and y^{d} are highly correlated ($\rho \approx 1$) or anti-correlated ($\rho \approx -1$), the straight-line fit is much better than the constant fit. *Hint.* From (13.3) we have

$$\hat{y}^{\mathrm{d}} - y^{\mathrm{d}} = \rho\frac{\mathbf{std}(y^{\mathrm{d}})}{\mathbf{std}(x^{\mathrm{d}})}(x^{\mathrm{d}} - \mathbf{avg}(x^{\mathrm{d}})\mathbf{1}) - (y^{\mathrm{d}} - \mathbf{avg}(y^{\mathrm{d}})\mathbf{1}).$$

Expand the norm squared of this expression, and use

$$\rho = \frac{(x^{\mathrm{d}} - \mathbf{avg}(x^{\mathrm{d}})\mathbf{1})^T(y^{\mathrm{d}} - \mathbf{avg}(y^{\mathrm{d}})\mathbf{1})}{\|x^{\mathrm{d}} - \mathbf{avg}(x^{\mathrm{d}})\mathbf{1}\|\|y^{\mathrm{d}} - \mathbf{avg}(y^{\mathrm{d}})\mathbf{1}\|}.$$

Solution. The residual for the straight-line fit is

$$\begin{aligned}\hat{y}^{\mathrm{d}} - y^{\mathrm{d}} &= \mathbf{avg}(y^{\mathrm{d}})\mathbf{1} + \rho\frac{\mathbf{std}(y^{\mathrm{d}})}{\mathbf{std}(x^{\mathrm{d}})}(x^{\mathrm{d}} - \mathbf{avg}(x^{\mathrm{d}})\mathbf{1}) - y^{\mathrm{d}} \\ &= -\tilde{y} + \rho\frac{\mathbf{std}(y^{\mathrm{d}})}{\mathbf{std}(x^{\mathrm{d}})}\tilde{x},\end{aligned}$$

where $\tilde{x}$ and $\tilde{y}$ are the de-meaned versions of x^{d} and y^{d}. Its norm squared is

$$\begin{aligned}\|r^{\mathrm{d}}\|^2 &= \|\tilde{y}\|^2 - 2\rho\frac{\mathbf{std}(y^{\mathrm{d}})}{\mathbf{std}(x^{\mathrm{d}})}\tilde{x}^T\tilde{y} + \rho^2\frac{\mathbf{std}(y^{\mathrm{d}})^2}{\mathbf{std}(x^{\mathrm{d}})^2}\|\tilde{x}\|^2 \\ &= N\,\mathbf{std}(y^{\mathrm{d}})^2 - 2\rho^2\frac{\mathbf{std}(y^{\mathrm{d}})}{\mathbf{std}(x^{\mathrm{d}})}\|\tilde{x}\|\|\tilde{y}\| + \rho^2\,\mathbf{std}(y^{\mathrm{d}})^2N \\ &= N\,\mathbf{std}(y^{\mathrm{d}})^2 - 2N\rho^2\,\mathbf{std}(y^{\mathrm{d}})^2 + \rho^2\,\mathbf{std}(y^{\mathrm{d}})^2N \\ &= N\,\mathbf{std}(y^{\mathrm{d}})^2(1-\rho^2),\end{aligned}$$

using $\tilde{x}^T\tilde{y} = \rho\|\tilde{x}\|\|\tilde{y}\|$ and $\|\tilde{x}\|^2 = N\,\mathbf{std}(x^{\mathrm{d}})^2$. Dividing by N and taking the squareroot we get

$$\mathbf{rms}(\hat{r}) = \mathbf{std}(y^{\mathrm{d}})\sqrt{1-\rho^2}.$$

13.2 *Regression to the mean.* Consider a data set in which the (scalar) $x^{(i)}$ is the parent's height (average of mother's and father's height), and $y^{(i)}$ is their child's height. Assume that over the data set the parent and child heights have the same mean value μ, and the same standard deviation σ. We will also assume that the correlation coefficient ρ between parent and child heights is (strictly) between zero and one. (These assumptions hold, at least approximately, in real data sets that are large enough.) Consider the simple straight-line fit or regression model given by (13.3), which predicts a child's height from the parent's height. Show that this prediction of the child's height lies (strictly) between the parent's height and the mean height μ (unless the parent's height happens to be exactly the mean μ). For example, if the parents are tall, *i.e.*, have height above the mean, we predict that their child will be shorter, but still tall. This phenomenon, called *regression to the mean*, was first observed by the early statistician Sir Francis Galton (who indeed, studied a data set of parent's and child's heights).

Solution. From (13.3) we have

$$\hat{f}(x) = \mu + \rho(x - \mu) = (1 - \rho)\mu + \rho x.$$

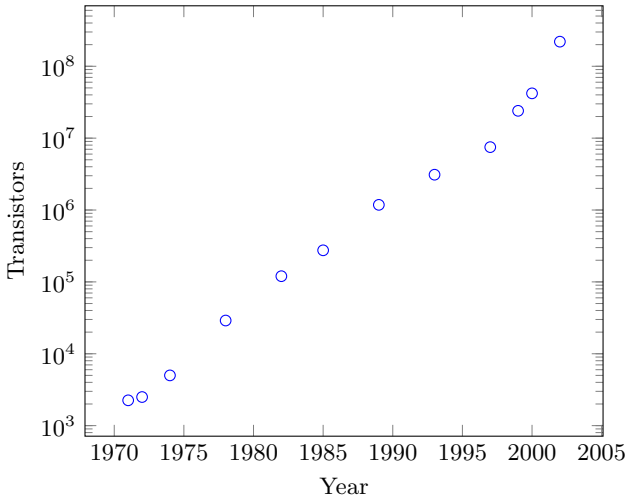
Since ρ satisfies $0 < \rho < 1$, we see that the predicted child's height is a convex combination of the mean μ and parent's height x. But a convex combination of any two numbers lies between the two numbers. When the parents are tall, *i.e.*, $x > \mu$, we have

$$\mu < \hat{f}(x) < x,$$

i.e., the predicted child's height is above average but also less than the parent's height.

13.3 *Moore's law.* The figure and table below show the number of transistors N in 13 microprocessors, and the year of their introduction.

Year	Transistors
1971	2,250
1972	2,500
1974	5,000
1978	29,000
1982	120,000
1985	275,000
1989	1,180,000
1993	3,100,000
1997	7,500,000
1999	24,000,000
2000	42,000,000
2002	220,000,000
2003	410,000,000



The plot gives the number of transistors on a logarithmic scale. Find the least squares straight-line fit of the data using the model

$$\log_{10} N \approx \theta_1 + \theta_2(t - 1970),$$

where t is the year and N is the number of transistors. Note that θ_1 is the model's prediction of the log of the number of transistors in 1970, and 10^{θ_2} gives the model's prediction of the fractional increase in number of transistors per year.

(a) Find the coefficients θ_1 and θ_2 that minimize the RMS error on the data, and give the RMS error on the data. Plot the model you find along with the data points.

(b) Use your model to predict the number of transistors in a microprocessor introduced in 2015. Compare the prediction to the IBM Z13 microprocessor, released in 2015, which has around 4×10^9 transistors.

(c) Compare your result with Moore's law, which states that the number of transistors per integrated circuit roughly doubles every one and a half to two years.

The computer scientist and Intel corporation co-founder Gordon Moore formulated the law that bears his name in a magazine article published in 1965.

Solution.

(a) We minimize the RMS error by minimizing the sum of the squares of the prediction errors,

$$\sum_{k=1}^{13}(\log_{10} n_k - \theta_1 - (t_k - 1970)\theta_2)^2 = \|A\theta - b\|^2,$$

where

$$A = \begin{bmatrix} 1 & t_1 - 1970 \\ 1 & t_2 - 1970 \\ \vdots & \vdots \\ 1 & t_{13} - 1970 \end{bmatrix}, \qquad b = \begin{bmatrix} \log_{10} N_1 \\ \log_{10} N_2 \\ \vdots \\ \log_{10} N_{13} \end{bmatrix}.$$

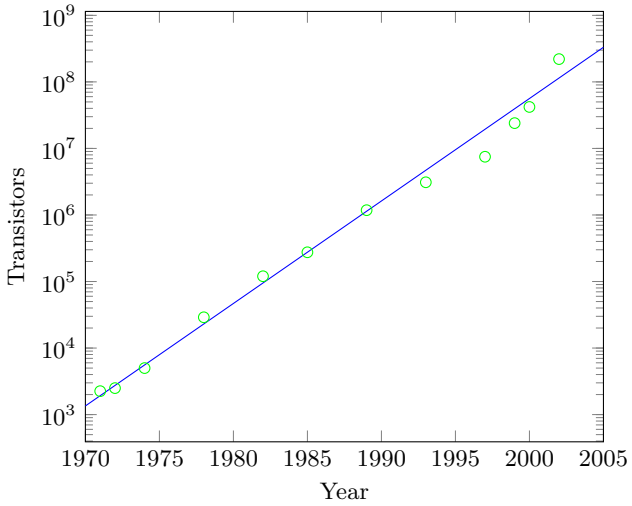
The solution is

$$\hat{\theta}_1 = 3.13, \qquad \hat{\theta}_2 = 0.154.$$

The RMS error of this model is

$$\frac{1}{\sqrt{13}}\|A\hat{\theta} - b\| = 0.20.$$

This means that we can expect our prediction of $\log_{10} N$ to typically be off by around 0.20. This corresponds to a prediction typically off by around a factor of $10^{0.2} = 1.6$. The straight-line fit is shown in the following figure.



(b) The predicted number of transistors in 2015 is

$$10^{\theta_1 + \theta_2(2015-1970)} \approx 1.14 \times 10^{10}$$

This prediction is about a factor of 3 off from the IBM Z13 processor. That is around two standard deviations off from the prediction, which is reasonable. (Although in general we would not expect extrapolations to have the same error as observed on the training data set.)

(c) In our model the number of transistors doubles approximately every $\log_{10} 2/\theta_2 = 1.95$ years, which is consistent with Moore's law.

13.4 *Asset α and β and market correlation.* Suppose the T-vectors $r^{\text{ind}} = (r_1^{\text{ind}}, \ldots, r_T^{\text{ind}})$ and $r^{\text{mkt}} = (r_1^{\text{mkt}}, \ldots, r_T^{\text{mkt}})$ are return time series for a specific asset and the whole market, as described on page 251. We let r^{rf} denote the risk-free interest rate, μ^{mkt} and σ^{mkt} the market return and risk (*i.e.*, $\mathbf{avg}(r^{\text{mkt}})$ and $\mathbf{std}(r^{\text{mkt}})$), and μ and σ the return and risk

of the asset (*i.e.*, $\mathbf{avg}(r^{\mathrm{ind}})$ and $\mathbf{std}(r^{\mathrm{ind}})$). Let ρ be the correlation coefficient between the market and asset return time series r^{mkt} and r^{ind}. Express the asset α and β in terms of r^{rf}, μ, σ, μ^{mkt}, σ^{mkt}, and ρ.

Solution. We have two formulas for the straight-line prediction of asset return, the one that defines α and β,

$$\hat{f}(x) = (r^{\mathrm{rf}} + \alpha) + \beta(x - \mu^{\mathrm{mkt}}),$$

and the generic formula (13.3),

$$\hat{f}(x) = \mathbf{avg}(y^{\mathrm{d}}) + \rho\frac{\mathbf{std}(y^{\mathrm{d}})}{\mathbf{std}(x^{\mathrm{d}})}(x - \mathbf{avg}(x^{\mathrm{d}}))$$

with $x^{\mathrm{d}} = r^{\mathrm{mkt}}$ and $y^{\mathrm{d}} = r^{\mathrm{ind}}$. Comparing coefficients we see that $\mu^{\mathrm{mkt}} = \mathbf{avg}(r^{\mathrm{mkt}})$ and

$$\alpha = \mathbf{avg}(r^{\mathrm{ind}}) - r^{\mathrm{rf}}, \qquad \beta = \rho\frac{\sigma}{\sigma^{\mathrm{mkt}}}.$$

13.5 *Polynomial model with multiple features.* The idea of polynomial models can be extended from the case discussed on page 255 where there is only one feature. In this exercise we consider a quadratic (degree two) model with 3 features, *i.e.*, x is a 3-vector. This has the form

$$\hat{f}(x) = a + b_1x_1 + b_2x_2 + b_3x_3 + c_1x_1^2 + c_2x_2^2 + c_3x_3^2 + c_4x_1x_2 + c_5x_1x_3 + c_6x_2x_3,$$

where the scalar a, 3-vector b, and 6-vector c are the zeroth, first, and second order coefficients in the model. Put this model into our general linear in the parameters form, by giving p, and the basis functions $f_1, \ldots, f_p$ (which map 2-vectors to scalars).

Solution. One choice is

$$\begin{aligned}
f_1(x) &= 1,\\
f_2(x) &= x_1,\\
f_3(x) &= x_2,\\
f_4(x) &= x_3,\\
f_5(x) &= x_1^2,\\
f_6(x) &= x_2^2,\\
f_7(x) &= x_3^2,\\
f_8(x) &= x_1x_2,\\
f_9(x) &= x_1x_3,\\
f_{10}(x) &= x_2x_3,
\end{aligned}$$

with $p = 10$. The coefficient 10-vector θ is $\theta = (a, b, c)$.

13.6 *Average prediction error.* Consider a data fitting problem, with first basis function $\phi_1(x) = 1$, and data set $x^{(1)}, \ldots, x^{(N)}$, $y^{(1)}, \ldots, y^{(N)}$. Assume the matrix A in (13.1) has linearly independent columns and let $\hat{\theta}$ denote the parameter values that minimize the mean square prediction error over the data set. Let the N-vector $\hat{r}^{\mathrm{d}}$ denote the prediction errors using the optimal model parameter $\hat{\theta}$. Show that $\mathbf{avg}(\hat{r}^{\mathrm{d}}) = 0$. In other words: With the least squares fit, the mean of the prediction errors over the data set is zero. *Hint.* Use the orthogonality principle (12.9), with $z = e_1$.

Solution. Since $Ae_1 = \mathbf{1}$, it follows from the orthogonality principle with $z = e_1$ that $\mathbf{1} \perp \hat{r}$. Therefore $\mathbf{avg}(\hat{r}) = (\mathbf{1}^T\hat{r})/N = 0$.

13.7 *Data matrix in auto-regressive time series model.* An auto-regressive model with memory M is fit by minimizing the sum of the squares of the predictions errors on a data set with T samples, $z_1, \ldots, z_T$, as described on page 259. Find the matrix A and vector y for which $\|A\beta - y\|^2$ gives the sum of the squares of the prediction errors. Show that A is a Toeplitz matrix (see page 138), *i.e.*, entries A_{ij} with the same value of $i-j$ are the same.

Solution.

$$A = \begin{bmatrix} z_M & z_{M-1} & z_{M-2} & \cdots & z_1 \\ z_{M+1} & z_M & z_{M-1} & \cdots & z_2 \\ z_{M+2} & z_{M+1} & z_M & \cdots & z_2 \\ \vdots & \vdots & \vdots & & \vdots \\ z_{T-1} & z_{T-2} & z_{T-3} & \cdots & z_{T-M} \end{bmatrix}, \qquad b = \begin{bmatrix} z_{M+1} \\ z_{M+2} \\ z_{M+3} \\ \vdots \\ z_T \end{bmatrix}.$$

13.8 *Fitting an input-output convolution system.* Let $u_1, \ldots, u_T$ and $y_1, \ldots, y_T$ be observed input and output time series for a system that is thought to be an input-output convolution system, meaning

$$y_t \approx \hat{y}_t = \sum_{j=1}^{n} h_j u_{t-j+1}, \quad t = 1, \ldots, T,$$

where we interpret u_t as zero for $t \leq 0$. Here the n-vector h is the system impulse response; see page 140. This model of the relation between the input and output time series is also called a *moving average* (MA) model. Find a matrix A and vector b for which

$$\|Ah - b\|^2 = (y_1 - \hat{y}_1)^2 + \cdots + (y_T - \hat{y}_T)^2.$$

Show that A is Toeplitz. (See page 138.)

Solution.

$$A = \begin{bmatrix} u_1 & 0 & 0 & \cdots & 0 \\ u_2 & u_1 & 0 & \cdots & 0 \\ u_3 & u_2 & u_1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ u_n & u_{n-1} & u_{n-2} & \cdots & u_1 \\ u_{n+1} & u_n & u_{n-1} & \cdots & u_2 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ u_T & u_{T-1} & u_{T-2} & \cdots & u_{T-N+1} \end{bmatrix}, \qquad b = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_T \end{bmatrix}.$$

13.9 *Conclusions from 5-fold cross-validation.* You have developed a regression model for predicting a scalar outcome y from a feature vector x of dimension 20, using a collection of $N = 600$ data points. The mean of the outcome variable y across the given data is 1.85, and its standard deviation is 0.32. After running 5-fold cross-validation we get the following RMS test errors (based on forming a model based on the data excluding fold i, and testing it on fold i).

Fold excluded	RMS test error
1	0.13
2	0.11
3	0.09
4	0.11
5	0.14

(a) How would you expect your model to do on new, unseen (but similar) data? Respond briefly and justify your response.

(b) A co-worker observes that the regression model parameters found in the 5 different folds are quite close, but not the same. He says that for the production system, you should use the regression model parameters found when you excluded fold 3 from the original data, since it achieved the best RMS test error. Comment briefly.

Solution.

(a) We observe that the RMS test errors are similar across the folds, so it is reasonable to expect the model to generalize well on similar data, with RMS error about 0.12 or so. We cannot guarantee this prediction; it will hold only to the extent that the new data is similar to the given data.

(b) This suggestion makes no sense, and is a bad idea. The model in fold 3 got slightly better RMS test error by luck, and there is no reason to think it is a better model than the other 4 models.

13.10 *Augmenting features with the average.* You are fitting a regression model $\hat{y} = x^T\beta + v$ to data, computing the model coefficients β and v using least squares. A friend suggests adding a new feature, which is the average of the original features. (That is, he suggests using the new feature vector $\tilde{x} = (x, \mathbf{avg}(x))$.) He explains that by adding this new feature, you might end up with a better model. (Of course, you would test the new model using validation.) Is this a good idea?

Solution. This is a bad idea. With the new feature the matrix A in (13.1) becomes the matrix

$$\tilde{A} = \begin{bmatrix} X^T & a \end{bmatrix},$$

where a is the vector with entries $a_i = \mathbf{avg}(x^{(i)})$. In other words, in $\tilde{A}$ the last column is $1/n$ times the sum of the other columns. In particular, the last column of $\tilde{A}$ is a linear combination of the other columns. It follows that the matrix $\tilde{A}$ does not have linearly independent columns, and the QR factorization will fail. This means that we cannot solve the associated least squares problem.

Even if we could solve the associated least squares problem, our results would not be any better. Any linear combination of the columns of $\tilde{A}$ is also a linear combination of the columns of A, and it follows that we cannot find a linear combination of the columns of $\tilde{A}$ that is closer to b than the optimal linear combination of the columns of A.

13.11 *Interpreting model fitting results.* Five different models are fit using the same training data set, and tested on the same (separate) test set (which has the same size as the training set). The RMS prediction errors for each model, on the training and test sets, are reported below. Comment briefly on the results for each model. You might mention whether the model's predictions are good or bad, whether it is likely to generalize to unseen data, or whether it is over-fit. You are also welcome to say that you don't believe the results, or think the reported numbers are fishy.

Model	Train RMS	Test RMS
A	1.355	1.423
B	9.760	9.165
C	5.033	0.889
D	0.211	5.072
E	0.633	0.633

Solution.

(a) This is a good model, and likely will generalize.

(b) This is a bad model, but will likely generalize.

(c) Something is wrong, or you are outrageously lucky. Probably the former.

(d) The model is over-fit.

(e) These results are suspicious, since it's unlikely that the train and test RMS errors would be so close. For example, maybe the model was accidentally tested on the training set. If the numbers are correct, then this is a very good model, and would likely generalize.

13.12 *Standardizing Boolean features.* (See page 269.) Suppose that the N-vector x gives the value of a (scalar) Boolean feature across a set of N examples. (Boolean means that each x_i has the value 0 or 1. This might represent the presence or absence of a symptom, or whether or not a day is a holiday.) How do we standardize such a feature? Express your answer in terms of p, the fraction of x_i that have the value 1. (You can assume that $p > 0$ and $p < 1$; otherwise the feature is constant.)

Solution. The mean value of x_i across the data set is p. The standard deviation of x_i across the data sample of size N is given by

$$\begin{aligned}
\mathbf{std}(x)^2 &= \frac{1}{N}\sum_{i=1}^{N}(x_i-p)^2 \\
&= \frac{1}{N}\left(\sum_{i=1}^{N}x_i^2 - 2p\sum_{i=1}^{N}x_i + Np^2\right) \\
&= p - 2p^2 + p^2 \\
&= p(1-p).
\end{aligned}$$

So $\mathbf{std}(x) = \sqrt{p(1-p)}$. The standardized feature x_i is then

$$z_i = \frac{x_i - p}{\sqrt{p(1-p)}} = \begin{cases} -\sqrt{p/(1-p)} & x_i = 0 \\ +\sqrt{(1-p)/p} & x_i = 1. \end{cases}$$

13.13 *Interaction model with Boolean features.* Consider a data fitting problem in which all n original features are Boolean, *i.e.*, entries of x have the value 0 or 1. These features could be the results of Boolean tests for a patient (or absence or presence of symptoms), or a person's response to a survey with yes/no questions. We wish to use these to predict an outcome, the number y. Our model will include a constant feature 1, the original n Boolean features, and all *interaction terms*, which have the form x_ix_j where $1 \le i < j \le n$.

(a) What is p, the total number of basis functions, in this model? Explicitly list the basis functions for the case $n = 3$. You can decide their order. *Hint.* To count the number of pairs i, j that satisfy $1 \le i < j \le n$, use equation (5.7).

(b) Interpret (together) the following three coefficients of θ: the one associated with the original feature x_3; the one associated with the original feature x_5; and the one associated with the new product feature x_3x_5. *Hint.* Consider the four possible values of the pair x_3, x_5.

Solution.

(a) The number of basis functions is

$$1 + n + \frac{n(n-1)}{2} = (n^2 + n + 2)/2.$$

This includes the feature 1, the n features $x_1, \ldots, x_n$, and the $n(n-1)/2$ functions x_ix_j, for $1 \le i < j \le n$.

To see that $n(n-1)/2$ is the number of interaction functions, we note that for $j = 2, \ldots, n$, we have the products $x_1x_j, \ldots, x_{j-1}x_j$. There are $j-1$ of these. So the total is

$$\sum_{j=2}^{n}(j-1) = 1 + 2 + 3 + \cdots + (n-1).$$

Using the equation (5.7), this is $n(n-1)/2$.
For $n = 3$ there are seven basis functions:

$$1, \quad x_1, \quad x_2, \quad x_3, \quad x_1x_2, \quad x_1x_3, \quad x_2x_3.$$

(b) Suppose the coefficients of the features x_3, x_5, x_3x_5 are θ_1, θ_2, θ_3, respectively. The contribution $\theta_1x_3 + \theta_2x_5 + \theta_3x_3x_5$ of the three features to the prediction is given in the following table.

x_3	x_5	$\theta_1x_3 + \theta_2x_5 + \theta_3x_3x_5$
0	0	0
1	0	θ_1
0	1	θ_2
1	1	$\theta_1 + \theta_2 + \theta_3$

Thus θ_1 is the amount the prediction goes up when feature 3 is 1 (true) but feature 5 is 0 (false). Similarly, θ_2 is the amount the prediction goes up when feature 3 is 1 but feature 5 is 0. The more interesting one to interpret is θ_3. It is the amount that the prediction goes up if features 3 and 5 are both 1, above and beyond their separate effects. If $\theta_3 > 0$, it means that the combination of the two features is 'more powerful' (in terms of the prediction) than their separate effects; if $\theta_3 < 0$, they are less powerful together than the sum of their separate effects.

13.14 *Least squares timing.* A computer takes around one second to fit a regression model (using least squares) with 20 parameters using 10^6 data points.

(a) About how long do you guess it will take the same computer to fit the same 20-parameter model using 10^7 data points (*i.e.*, 10× more data points)?

(b) About how long do you guess it will take the same computer to fit a 200-parameter model using 10^6 data points (*i.e.*, 10× more model parameters)?

Solution. The flop count for least squares with m data points and n parameters is $2mn^2$. This means that it grows linearly in the number of data points, and quadratically in the number of parameters. This gives us our answers.

(a) About 10 seconds.

(b) About 100 seconds.

13.15 *Estimating a matrix.* Suppose that the n-vector x and the m-vector y are thought to be approximately related by a linear function, *i.e.*, $y \approx Ax$, where A is an $m \times n$ matrix. We do not know the matrix A, but we do have observed data,

$$x^{(1)}, \ldots, x^{(N)}, \qquad y^{(1)}, \ldots, y^{(N)}.$$

We can estimate or guess the matrix A by choosing it to minimize

$$\sum_{i=1}^{N} \|Ax^{(i)} - y^{(i)}\|^2 = \|AX - Y\|^2,$$

where $X = [x^{(1)} \ \cdots \ x^{(N)}]$ and $Y = [y^{(1)} \ \cdots \ y^{(N)}]$. We denote this *least squares estimate* as $\hat{A}$. (The notation here can be confusing, since X and Y are known, and A is to be found; it is more conventional to have symbols near the beginning of the alphabet, like A, denote known quantities, and symbols near the end, like X and Y, denote variables or unknowns.)

(a) Show that $\hat{A} = YX^\dagger$, assuming the rows of X are linearly independent. *Hint.* Use $\|AX - Y\|^2 = \|X^TA^T - Y^T\|^2$, which turns the problem into a matrix least squares problem; see page 233.

(b) Suggest a good way to compute $\hat{A}$, and give the complexity in terms of n, m, and N.

Solution.

(a) Taking transposes as suggested, we find that $W = A^T$ is the $n \times m$ matrix that minimizes

$$\|X^T W - Y^T\|^2.$$

Let w_k be the kth column of W and y_k be the kth column of Y^T. Then

$$\|X^T W - Y^T\|^2 = \sum_{k=1}^{m} \|X^T w_k - y_k\|^2.$$

Since each w_k appears in only one term in the sum, the optimal w_k is the vector that minimizes $\|X^T w_k - y_k\|^2$, *i.e.*,

$$\hat{w}_k = (XX^T)^{-1} X y_k.$$

Therefore

$$\begin{aligned}
\hat{A} &= \hat{W}^T \\
&= \begin{bmatrix} \hat{w}_1 & \hat{w}_2 & \cdots & \hat{w}_m \end{bmatrix}^T \\
&= \begin{bmatrix} (XX^T)^{-1}Xy_1 & (XX^T)^{-1}Xy_2 & \cdots & (XX^T)^{-1}Xy_m \end{bmatrix}^T \\
&= \left((XX^T)^{-1}XY^T\right)^T \\
&= YX^T(XX^T)^{-1} \\
&= YX^{\dagger}.
\end{aligned}$$

(b) We use the QR factorization $X^T = QR$. The pseudo-inverse of X is

$$X^T(XX^T)^{-1} = QR(R^TR)^{-1} = QRR^{-1}R^{-T} = QR^{-T}$$

and therefore the solution of the problem can be computed from the QR factorization using the formula $\hat{A} = YQR^{-T}$. This can be computed in three steps.

- QR factorization $X^T = QR$ of the $N \times n$ matrix X^T. ($2Nn^2$ flops.)
- Matrix-matrix product $Z = YQ$ of the $m \times N$ matrix Y and the $N \times n$ matrix Q. ($2mnN$ flops.)
- Compute the $m \times n$ matrix $\hat{A} = ZR^{-T}$ by solving the matrix equation $R^T\hat{A}^T = Z^T$, column by column, via forward substitution. (mn^2 flops.)

The total is $2Nn^2 + 2mnN + mn^2$ flops.

13.16 *Relative fitting error and error fitting the logarithm.* (See page 259.) The relative fitting error between a positive outcome y and its positive prediction $\hat{y}$ is given by $\eta = \max\{\hat{y}/y, y/\hat{y}\} - 1$. (This is often given as a percentage.) Let r be the residual between their logarithms, $r = \log y - \log \hat{y}$. Show that $\eta = e^{|r|} - 1$.

Solution. Let $w = \log y$ and $\hat{w} = \log \hat{y}$. Suppose that the error in fitting w is r, *i.e.*, $\hat{w} - w = r$. Taking the exponential we get

$$e^r = e^{\hat{w}-w} = e^{\hat{w}}/e^w = \hat{y}/y.$$

Likewise we get

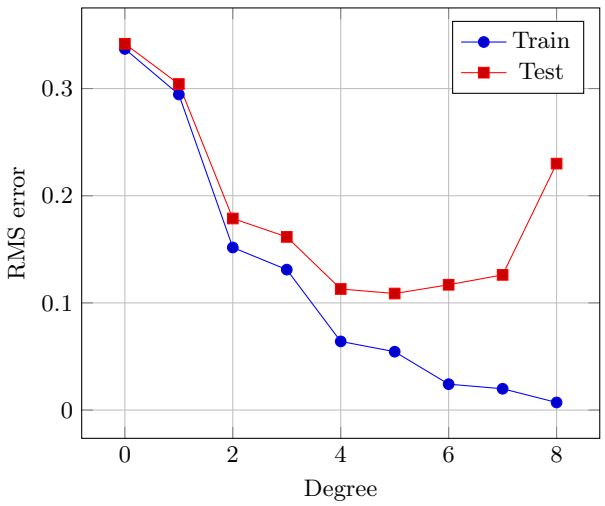
$$e^{-r} = e^{w-\hat{w}} = y/\hat{y}.$$

So $\max\{\hat{y}/y, y/\hat{y}\} = \max\{e^r, e^{-r}\} = e^{|r|}$. Subtracting one from both sides gives the equation above.

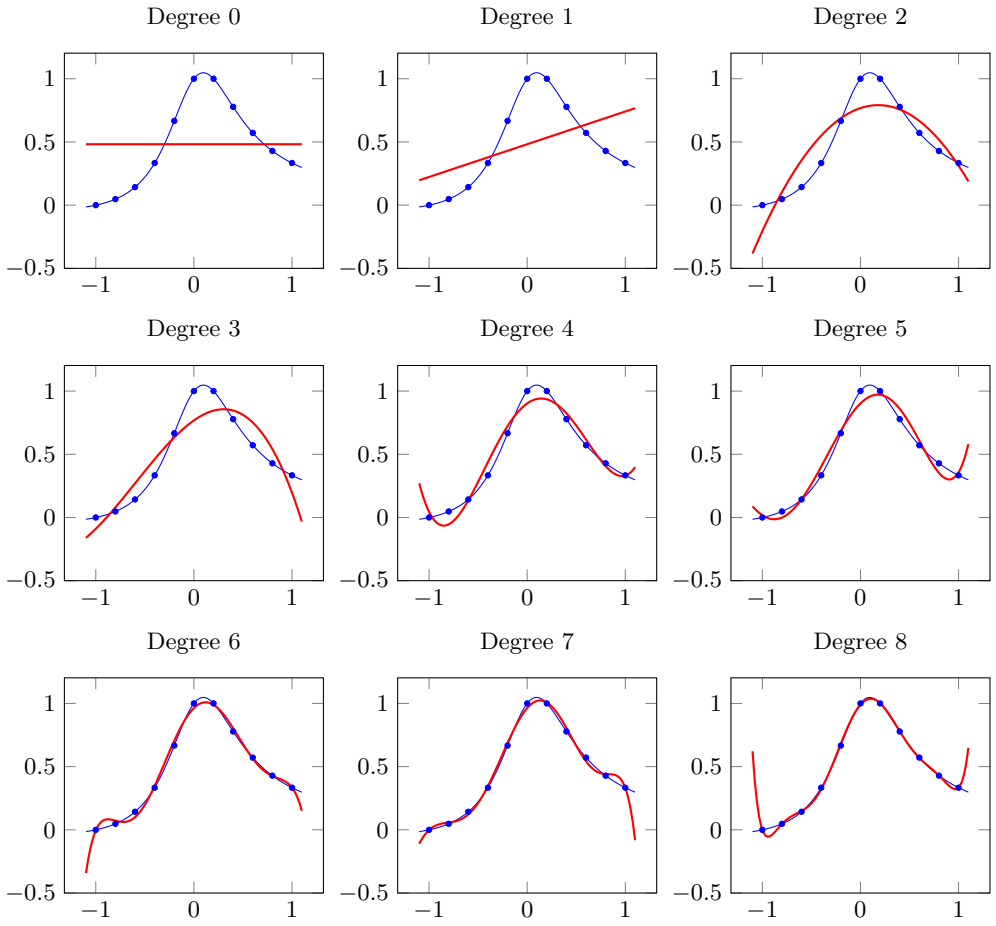
13.17 *Fitting a rational function with a polynomial.* Let $x_1, \ldots, x_{11}$ be 11 points uniformly spaced in the interval $[-1, 1]$. (This means $x_i = -1.0 + 0.2(i-1)$ for $i = 1, \ldots, 11$.) Take $y_i = (1 + x_i)/(1 + 5x_i^2)$, for $i = 1, \ldots, 11$. Find the least squares fit of polynomials of degree $0, 1, \ldots, 8$ to these points. Plot the fitting polynomials, and the true function $y = (1 + x)/(1 + 5x^2)$, over the interval $[-1.1, 1.1]$ (say, using 100 points). Note that the interval for the plot, $[-1.1, 1.1]$, extends a bit outside the range of the data used to fit the polynomials, $[-1, 1]$; this gives us an idea of how well the polynomial fits can extrapolate.

Generate a test data set by choosing $u_1, \ldots, u_{10}$ uniformly spaced over $[-1.1, 1.1]$, with $v_i = (1+u_i)/(1+5u_i^2)$. Plot the RMS error of the polynomial fits found above on this test data set. On the same plot, show the RMS error of the polynomial fits on the training data set. Suggest a reasonable value for the degree of the polynomial fit, based on the RMS fits on the training and test data. *Remark.* There is no practical reason to fit a rational function with a polynomial. This exercise is only meant to illustrate the ideas of fitting with different basis functions, over-fit, and validation with a test data set.

Solution. The first figure shows the RMS errors on the training and test data set. This suggests using degree four or five.



The second figure shows the nine fitted polynomials.



13.18 *Vector auto-regressive model.* The auto-regressive time series model (see page 259) can be extended to handle times series $z_1, z_2, \ldots$ where z_t are n-vectors. This arises in applications such as econometrics, where the entries of the vector give different economic quantities. The vector auto-regressive model (already mentioned on page 164) has the same form as a scalar auto-regressive model,

$$\hat{z}_{t+1} = \beta_1 z_t + \cdots + \beta_M z_{t-M+1}, \quad t = M, M+1, \ldots$$

where M is the *memory* of the model; the difference here is that the model parameters $\beta_1, \ldots, \beta_M$ are all $n \times n$ matrices. The total number of (scalar) parameters in the vector auto-regressive model is Mn^2. These parameters are chosen to minimize the sum of the squared norms of the prediction errors over a data set $z_1, \ldots, z_T$,

$$\|z_{M+1} - \hat{z}_{M+1}\|^2 + \cdots + \|z_T - \hat{z}_T\|^2.$$

(a) Give an interpretation of $(\beta_2)_{13}$. What would it mean if this coefficient is very small?

(b) Fitting a vector auto-regressive model can be expressed as a matrix least squares problem (see page 233), *i.e.*, the problem of choosing the matrix X to minimize $\|AX - B\|^2$, where A and B are matrices. Find the matrices A and B. You can consider the case $M = 2$. (It will be clear how to generalize your answer to larger values of M.)

Hints. Use the $(2n) \times n$ matrix variable $X = \begin{bmatrix} \beta_1 & \beta_2 \end{bmatrix}^T$, and right-hand side ($(T-2) \times n$ matrix) $B = \begin{bmatrix} z_3 & \cdots & z_T \end{bmatrix}^T$. Your job is to find the matrix A so that $\|AX - B\|^2$ is the sum of the squared norms of the prediction errors.

Solution.

(a) $(\beta_2)_{13}$ is the coefficient of $(z_{t-1})_3$ in $(\hat{z}_{t+1})_1$. It gives the factor by which the first component of our prediction of the next vector value depends on the third component of the previous vector value. (Whew!) If this coefficient were very small, it means that our prediction of the first entry of the next time series element does not very much depend on the third entry of z_{t-1}.

(b) We follow the hints, using the given X and B. The matrix AX should be the matrix of predictions, following the pattern in B,

$$AX = \begin{bmatrix} \hat{z}_3 & \cdots & \hat{z}_T \end{bmatrix}^T = \begin{bmatrix} z_3^T \\ \vdots \\ z_T^T \end{bmatrix}.$$

In this case $\|AX - B\|^2$ is the sum of the squared norms of the prediction errors.

Now let's find the matrix A for which AX is the matrix of predictions as above. We want to find A so that

$$A \begin{bmatrix} \beta_1^T \\ \beta_2^T \end{bmatrix} = \begin{bmatrix} \hat{z}_3^T \\ \vdots \\ \hat{z}_T^T \end{bmatrix}.$$

Matching coefficients we find that A must be

$$A = \begin{bmatrix} z_2^T & z_1^T \\ \vdots & \vdots \\ z_{T-2}^T & z_{T-1}^T \end{bmatrix}.$$

So, fitting a vector AR model can be expressed as the matrix least squares problem of minimizing

$$\left\| \begin{bmatrix} z_2^T & z_1^T \\ \vdots & \vdots \\ z_{T-2}^T & z_{T-1}^T \end{bmatrix} \begin{bmatrix} \beta_1^T \\ \beta_2^T \end{bmatrix} - \begin{bmatrix} z_3^T \\ \vdots \\ z_T^T \end{bmatrix} \right\|^2.$$

13.19 *Sum of sinusoids time series model.* Suppose that $z_1, z_2, \ldots$ is a time series. A very common approximation of the time series is as a sum of K *sinusoids*

$$z_t \approx \hat{z}_t = \sum_{k=1}^{K} a_k \cos(\omega_k t - \phi_k), \quad t = 1, 2, \ldots.$$

The kth term in this sum is called a *sinusoid signal*. The coefficient $a_k \geq 0$ is called the *amplitude*, $\omega_k > 0$ is called the *frequency*, and ϕ_k is called the *phase* of the kth sinusoid. (The phase is usually chosen to lie in the range from $-\pi$ to π.) In many applications the frequencies are multiples of ω_1, *i.e.*, $\omega_k = k\omega_1$ for $k = 2, \ldots, K$, in which case the approximation is called a *Fourier approximation*, named for the mathematician Jean-Baptiste Joseph Fourier.

Suppose you have observed the values $z_1, \ldots, z_T$, and wish to choose the sinusoid amplitudes $a_1, \ldots, a_K$ and phases $\phi_1, \ldots, \phi_K$ so as to minimize the RMS value of the approximation error $(\hat{z}_1 - z_1, \ldots, \hat{z}_T - z_T)$. (We assume that the frequencies are given.) Explain how to solve this using least squares model fitting.

Hint. A sinusoid with amplitude a, frequency ω, and phase ϕ can be described by its cosine and sine coefficients α and β, where

$$a\cos(\omega t - \phi) = \alpha\cos(\omega t) + \beta\sin(\omega t),$$

where (using the cosine of sum formula) $\alpha = a\cos\phi$, $\beta = a\sin\phi$. We can recover the amplitude and phase from the cosine and sine coefficients as

$$a = \sqrt{\alpha^2 + \beta^2}, \qquad \phi = \arctan(\beta/\alpha).$$

Express the problem in terms of the cosine and sine coefficients.

Solution. Let z be the T-vector $(z_1, \ldots, z_T)$, and $\hat{z}$ the T-vector $(\hat{z}_1, \ldots, \hat{z}_T)$. Following the hint, we express $\hat{z}$ as

$$\hat{z}_t = \sum_{k=1}^{K} \left(\alpha_k \cos(\omega_k t) + \beta_k \sin(\omega_k t)\right), \quad t = 1, \ldots, T.$$

Our goal is to minimize $\|\hat{z} - z\|^2$. This is a function fitting problem with $p = 2k$ basis functions

$$f_i(t) = \cos(\omega_i t), \quad i = 1, \ldots, K, \qquad f_i(t) = \sin(\omega_{i-K} t), \quad i = K+1, \ldots, 2K.$$

13.20 *Fitting with continuous and discontinuous piecewise-linear functions.* Consider a fitting problem with $n = 1$, so $x^{(1)}, \ldots, x^{(N)}$ and $y^{(1)}, \ldots, y^{(N)}$ are numbers. We consider two types of closely related models. The first is a piecewise-linear model with knot points at -1 and 1, as described on page 256, and illustrated in figure 13.8. The second is a stratified model (see page 272), with three independent affine models, one for $x < -1$, one for $-1 \leq x \leq 1$, and one for $x > 1$. (In other words, we stratify on x taking low, middle, or high values.) Are these two models the same? Is one more general than the other? How many parameters does each model have? *Hint.* See problem title.

What can you say about the training set RMS error and test set RMS error that would be achieved using least squares with these two models?

Solution. They are not the same. In particular, the piecewise-linear model has $p = 4$ parameters, and the stratified model has 6 parameters (two parameters for each of three affine models). In the first case, the predictor is any *continuous* piecewise-linear function of x; in the second case, the predictor is any function that is affine in the three intervals of x. In other words, it is piecewise-linear but not necessarily continuous. This tells us that piecewise-linear models are a special case of the stratified affine models; any piecewise-linear model can also be represented by a stratified affine model. But not vice versa.

On the training data, the stratified affine model will achieve smaller (or the same) RMS error than the stratified affine model.

The last part of the question, about the test data, is partly a trick question. The correct answer is: We can nothing about which model will perform better on the test data set.

13.21 *Efficient cross-validation.* The cost of fitting a model with p basis functions and N data points (say, using QR factorization) is $2Np^2$ flops. In this exercise we explore the complexity of carrying out 10-fold cross-validation on the same data set. We divide the data set into 10 folds, each with $N/10$ data points. The naïve method is to fit 10 different models, each using 9 of the folds, using the QR factorization, which requires $10 \cdot 2(0.9)Np^2 = 18Np^2$ flops. (To evaluate each of these models on the remaining fold requires $2(N/10)p$ flops, which can be ignored compared to the cost of fitting the models.) So the naïve method of carrying out 10-fold cross-validation requires, not surprisingly, around $10\times$ the number of flops as fitting a single model.

The method below outlines another method to carry out 10-fold cross-validation. Give the total flop count for each step, keeping only the dominant terms, and compare the total cost

of the method to that of the naïve method. Let $A_1, \ldots, A_{10}$ denote the $(N/10) \times p$ blocks of the data matrix associated with the folds, and let $b_1, \ldots, b_{10}$ denote the right-hand sides in the least squares fitting problem.

(a) Form the Gram matrices $G_i = A_i^T A_i$ and the vectors $c_i = A_i^T b_i$.

(b) Form $G = G_1 + \cdots + G_{10}$ and $c = c_1 + \cdots + c_{10}$.

(c) For $k = 1, \ldots, 10$, compute $\theta_k = (G - G_k)^{-1}(c - c_k)$.

Solution.

(a) Forming G_i costs $2(N/10)p^2$ flops, so for all 10 folds, the cost is $2Np^2$ flops. Forming c_i is negligible compared to this.

(b) Forming G by adding G_i costs $9p^2$ flops, which can be ignored compared to the cost of step (a). Forming c is negligible.

(c) Forming $G - G_k$ for each k costs p^2 flops, so the total is $10p^2$ flops, negligible compared to $2Np^2$ flops in step (a). Forming $c - c_i$ is negligible. Computing θ_k costs $2(N/10)p^2$ flops, for a total of $2Np^2$ flops.

So the total complexity of the method is $4Np^2$, only twice the complexity of fitting a single model, and a factor 5 faster than carrying out cross-validation using the naïve method. In fact, using algorithms that exploit the symmetry of G_i the total effort can be brought down to around $2Np^2$ flops. In other words: 10-fold cross-validation can be done with the same complexity as fitting a single model.

13.22 *Prediction contests.* Several companies have run prediction contests open to the public. Netflix ran the best known contest, offering a $1M prize for the first prediction of user movie rating that beat their existing method RMS prediction error by 10% on a test set. The contests generally work like this (although there are several variations on this format, and most are more complicated). The company posts a public data set, that includes the regressors or features and the outcome for a large number of examples. They also post the features, but not the outcomes, for a (typically smaller) test data set. The contestants, usually teams with obscure names, submit predictions for the outcomes in the test set. Usually there is a limit on how many times, or how frequently, each team can submit a prediction on the test set. The company computes the RMS test set prediction error (say) for each submission. The teams' prediction performance is shown on a *leaderboard*, which lists the 100 or so best predictions in order.

Discuss such contests in terms of model validation. How should a team check a set of predictions before submitting it? What would happen if there were no limits on the number of predictions each team can submit? Suggest an obvious method (typically disallowed by the contest rules) for a team to get around the limit on prediction submissions. (And yes, it has been done.)

Solution. These contests formalize how predictors are really used. You get some data you can use to develop a model; you are judged by how well your predictor does on another set of data, where you don't know what the true outcomes are. At the least, a team should divide the original given data into a training set, the data they will use to fit the model, and a test set, a set of data on which the team (not the company) will test the model. This test set is the team's test set, not the official test set. If the team's test RMS error is good enough to get on the leaderboard, it would make sense to submit it. If the limit on submissions were not imposed, a team could submit a new set of predictions every second (or even faster). This would give an unfair advantage to a team that has the ability to generate many different predictions. The obvious way to get around this is to register a large number of fake teams, which are actually (but secretly) under the control of your team. The fake teams then submit their predictions separately.

Chapter 14

Least squares classification

Exercises

14.1 *Chebyshev bound.* Let $\tilde{f}(x)$ denote the continuous prediction of the Boolean outcome y, and $\hat{f}(x) = \mathbf{sign}(\tilde{f}(x))$ the actual classifier. Let σ denote the RMS error in the continuous prediction over some set of data, *i.e.*,

$$\sigma^2 = \frac{(\tilde{f}(x^{(1)}) - y^{(1)})^2 + \cdots + (\tilde{f}(x^{(N)}) - y^{(N)})^2}{N}.$$

Use the Chebyshev bound to argue that the error rate over this data set, *i.e.*, the fraction of data points for which $\hat{f}(x^{(i)}) \neq y^{(i)}$, is no more than σ^2, assuming $\sigma < 1$.

Remark. This bound on the error rate is usually quite bad; that is, the actual error rate in often much lower than this bound. But it does show that if the continuous prediction is good, then the classifier must perform well.

Solution. Let s be the N-vector with entries $s_i = \tilde{f}(x^{(i)}) - y^{(i)}$, so $\sigma = \mathbf{rms}(s)$. If a data point $x^{(i)}$ is misclassified then

$$|s_i| = |\tilde{f}(x^{(i)}) - y^{(i)}| \geq 1.$$

(Either $y^{(i)} = 1$ and $\tilde{f}(x^{(i)}) \leq 0$ or $y^{(i)} = -1$ and $\tilde{f}(x^{(i)}) \geq 0$.) Therefore $M \leq k$, if M denotes the number of misclassified data points and k the number of entries of s with $|s_i| \geq 1$. We have

$$\frac{M}{N} \leq \frac{k}{N} \leq \mathbf{rms}(s)^2 = \sigma^2.$$

The second inequality is the Chebyshev inequality (3.2) applied to the vector s and threshold $a = 1$.

14.2 *Interpreting the parameters in a regression classifier.* Consider a classifier of the form $\hat{y} = \mathbf{sign}(x^T\beta + v)$, where $\hat{y}$ is the prediction, the n-vector x is the feature vector, and the n-vector β and scalar v are the classifier parameters. We will assume here that $v \neq 0$ and $\beta \neq 0$. Evidently $\hat{y} = \mathbf{sign}(v)$ is the prediction when the feature vector x is zero. Show that when $\|x\| < |v|/\|\beta\|$, we have $\hat{y} = \mathbf{sign}(v)$. This shows that when all the features are *small* (*i.e.*, near zero), the classifier makes the prediction $\hat{y} = \mathbf{sign}(v)$. *Hint.* If two numbers a and b satisfy $|a| < |b|$, then $\mathbf{sign}(a + b) = \mathbf{sign}(b)$.

This means that we can interpret $\mathbf{sign}(v)$ as the classifier prediction when the features are small. The ratio $|v|/\|\beta\|$ tells us how big the feature vector must be before the classifier 'changes its mind' and predicts $\hat{y} = -\mathbf{sign}(v)$.

Solution. By the Cauchy–Schwarz inequality, $|x^T\beta| \leq \|x\|\|\beta\|$. If $\|x\| < |v|/\|\beta\|$, then

$$|x^T\beta| \leq \|x\|\|\beta\| < |v|.$$

Using the hint it follows that $\mathbf{sign}(x^T\beta + v) = \mathbf{sign}(v)$.

14.3 *Likert classifier.* A response to a question has the options *Strongly Disagree*, *Disagree*, *Neutral*, *Agree*, or *Strongly Agree*, encoded as $-2, -1, 0, 1, 2$, respectively. You wish to build a multi-class classifier that takes a feature vector x and predicts the response. A multi-class least squares classifier builds a separate (continuous) predictor for each response versus the others. Suggest a simpler classifier, based on one continuous regression model $\tilde{f}(x)$ that is fit to the numbers that code the responses, using least squares.

Solution. We simply round the continuous prediction $\tilde{f}(x)$ to the nearest of -2, -1, 0, 1, 2, and then predict that response.

14.4 *Multi-class classifier via matrix least squares.* Consider the least squares multi-class classifier described in §14.3, with a regression model $\tilde{f}_k(x) = x^T\beta_k$ for the one-versus-others classifiers. (We assume that the offset term is included using a constant feature.) Show that the coefficient vectors $\beta_1, \ldots, \beta_K$ can be found by solving the matrix least squares problem of minimizing $\|X^T\beta - Y\|^2$, where β is the $n \times K$ matrix with columns $\beta_1, \ldots, \beta_K$, and Y is an $N \times K$ matrix.

(a) Give Y, *i.e.*, describe its entries. What is the ith row of Y?

(b) Assuming the rows of X (*i.e.*, the data feature vectors) are linearly independent, show that the least squares estimate is given by $\hat{\beta} = (X^T)^\dagger Y$.

Solution.

(a) Define

$$Y_{ik} = \begin{cases} 1 & \text{data point } i \text{ belongs to class } k \\ -1 & \text{otherwise,} \end{cases}$$

and let y_k be the kth column of Y. Then

$$\|X^T\beta - Y\|^2 = \|X^T\beta_1 - y_1\|^2 + \cdots + \|X^T\beta_K - y_K\|^2.$$

The matrix least squares problem is equivalent to K ordinary least squares problems in which we minimize $\|X^T\beta_k - y_k\|^2$ over β_k. This is exactly the least squares problem for the Boolean class-k-versus-not-k classifier.

(b) This is an application of formula (12.12).

14.5 *List classifier.* Consider a multi-class classification problem with K classes. A standard multi-class classifier is a function $\hat{f}$ that returns a class (one of the labels $1, \ldots, K$), when given a feature n-vector x. We interpret $\hat{f}(x)$ as the classifier's guess of the class that corresponds to x. A *list classifier* returns a list of guesses, typically in order from 'most likely' to 'least likely'. For example, for a specific feature vector x, a list classifier might return $3, 6, 2$, meaning (roughly) that its top guess is class 3, its next guess is class 6, and its third guess is class 2. (The lists can have different lengths for different values of the feature vector.) How would you modify the least squares multi-class classifier described in §14.3.1 to create a list classifier? *Remark.* List classifiers are widely used in electronic communication systems, where the feature vector x is the received signal, and the class corresponds to which of K messages was sent. In this context they are called *list decoders*. List decoders produce a list of probable or likely messages, and allow a later processing stage to make the final decision or guess.

Solution. It's simple: First you create the K one-versus-others Boolean classifiers. As the most likely class for a given x, we return the k that corresponds to the largest value of $\tilde{f}_k(x)$; as the second most likely we return the value of k that corresponds to the second largest, and so on.

14.6 *Polynomial classifier with one feature.* Generate 200 points $x^{(1)}, \ldots, x^{(200)}$, uniformly spaced in the interval $[-1, 1]$, and take

$$y^{(i)} = \begin{cases} +1 & -0.5 \leq x^{(i)} < 0.1 \text{ or } 0.5 \leq x^{(i)} \\ -1 & \text{otherwise} \end{cases}$$

for $i = 1, \ldots, 200$. Fit polynomial least squares classifiers of degrees $0, \ldots, 8$ to this training data set.

(a) Evaluate the error rate on the training data set. Does the error rate decrease when you increase the degree?

(b) For each degree, plot the polynomial $\tilde{f}(x)$ and the classifier $\hat{f}(x) = \mathbf{sign}(\tilde{f}(x))$.

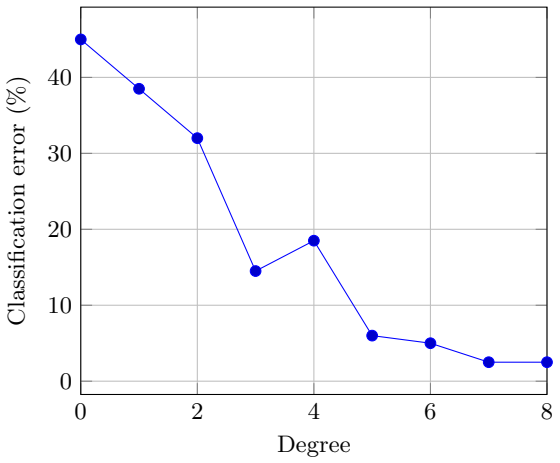
(c) It is possible to classify this data set perfectly using a classifier $\hat{f}(x) = \mathbf{sign}(\tilde{f}(x))$ and a cubic polynomial

$$\tilde{f}(x) = c(x + 0.5)(x - 0.1)(x - 0.5),$$

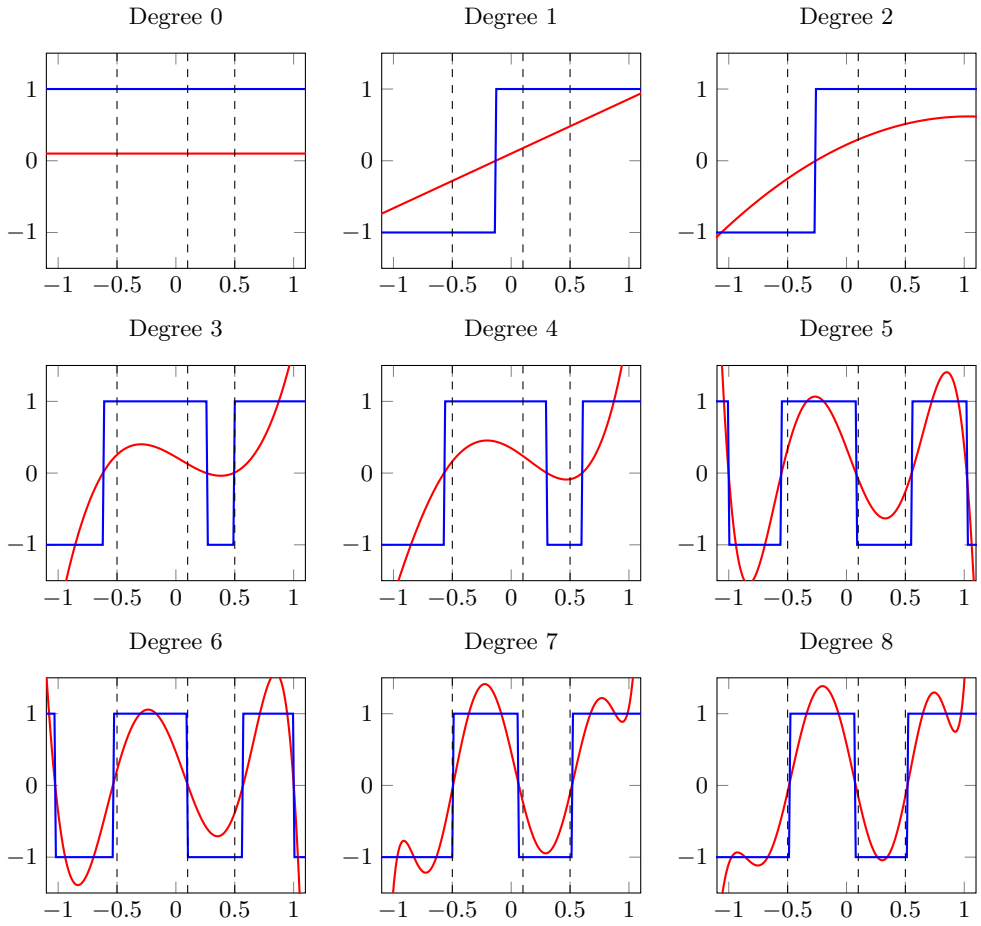
for any positive c. Compare this classifier with the least squares classifier of degree 3 that you found and explain why there is a difference.

Solution.

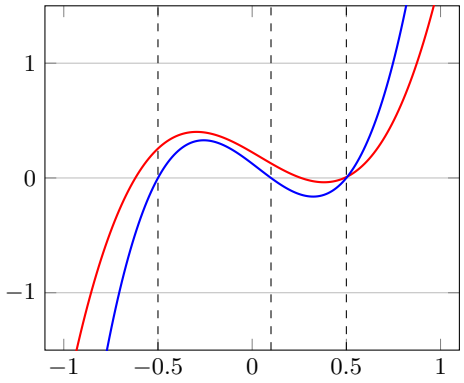
(a) The figure shows the error rate. We note that the error increases as we increase the degree from 3 to 4. This contrasts with RMS error in least squares data fitting, which never increases as we add basis functions.



(b) The second figure shows the polynomials $\tilde{f}(x)$ (in red) and the classifiers $\hat{f}(x) = \mathbf{sign}(\tilde{f}(x))$ (in blue).



(c) The least squares classifier is not equal to this optimal classifier. The figure shows the polynomial $5(x+0.5)(x-0.1)(x-0.5)$ in blue and the least squares solution of degree three in red.



The least squares classifier fits a polynomial $\tilde{f}(x)$ to the training data (x_i, y_i) by minimizing the squared error $\sum_i (\tilde{f}(x_i) - y_i)^2$. This is not the same as minimizing

the classification error $\sum_i (\mathbf{sign}(\tilde{f}(x_i)) - y_i)^2/(4N)$.

14.7 *Polynomial classifier with two features.* Generate 200 random 2-vectors $x^{(1)}, \ldots, x^{(200)}$ in a plane, from a standard normal distribution. Define

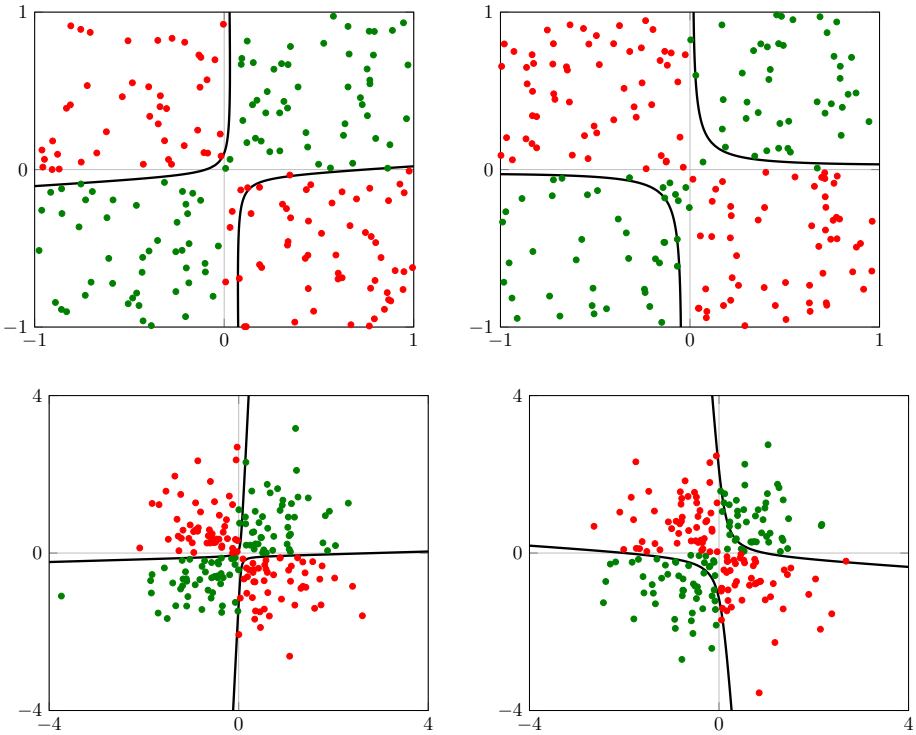
$$y^{(i)} = \begin{cases} +1 & x_1^{(i)} x_2^{(i)} \geq 0 \\ -1 & \text{otherwise} \end{cases}$$

for $i = 1, \ldots, 200$. In other words, $y^{(i)}$ is $+1$ when $x^{(i)}$ is in the first or third quadrant, and -1 otherwise. Fit a polynomial least squares classifier of degree 2 to the data set, *i.e.*, use a polynomial

$$\tilde{f}(x) = \theta_1 + \theta_2 x_1 + \theta_3 x_2 + \theta_4 x_1^2 + \theta_5 x_1 x_2 + \theta_6 x_2^2.$$

Give the error rate of the classifier. Show the regions in the plane where $\hat{f}(x) = 1$ and $\hat{f}(x) = -1$. Also compare the computed coefficients with the polynomial $\tilde{f}(x) = x_1 x_2$, which classifies the data points with zero error.

Solution. The figures shows the result for four different sets of points.



The heavy lines indicate the boundary between the regions where $\hat{f}(x) = 1$ and $\hat{f}(x) = -1$. For the first example, the computed coefficients are

$$\theta = (0.0117,\ 0.1047,\ -0.1161,\ -0.1622,\ 2.3108,\ 0.0358)$$

and 7 points are misclassified. For the second example, the computed coefficients are

$$\theta = (-0.0496,\ -0.0058,\ 0.0344,\ -0.0231,\ 2.3292,\ -0.0288)$$

and 12 points are misclassified. For the third example, the computed coefficients are

$$\theta = (0.0023,\ 0.0619,\ -0.0323,\ -0.0219,\ 0.6460,\ -0.0268)$$

and 3 points are misclassified. For the fourth example, the computed coefficients are

$$\theta = (-0.0946,\ 0.0500,\ -0.0364,\ 0.0490,\ 0.6440,\ 0.0385)$$

and 18 points are misclassified.

In all cases θ_5 is the largest coefficient, but the other coefficients are not zero.

14.8 *Author attribution.* Suppose that the N feature n-vectors $x^{(1)}, \ldots, x^{(N)}$ are word count histograms, and the labels $y^{(1)}, \ldots, y^{(N)}$ give the document authors (as one of $1, \ldots, K$). A classifier guesses which of the K authors wrote an unseen document, which is called *author attribution*. A least squares classifier using regression is fit to the data, resulting in the classifier

$$\hat{f}(x) = \operatorname*{argmax}_{k=1,\ldots,K}(x^T\beta_k + v_k).$$

For each author (*i.e.*, $k = 1, \ldots, K$) we find the ten largest (most positive) entries in the n-vector β_k and the ten smallest (most negative) entries. These correspond to two sets of ten words in the dictionary, for each author. Interpret these words, briefly, in English.

Solution. The vector β_k is used to classify whether the author is k, or one of the others. When $(\beta_k)_i$ is large (and positive) it means that word i is indicative, or often used by, author k, compared to the others. When $(\beta_k)_i$ is small (most negative) it means that word i is seldom used by author k, compared to the others.

The words in the first list are those that are most characteristic of author k, compared to the others. The words in the second list are those that are least characteristic of author k.

14.9 *Nearest neighbor interpretation of multi-class classifier.* We consider the least squares K-class classifier of §14.3.1. We associate with each data point the n-vector x, and the label or class, which is one of $1, \ldots, K$. If the class of the data point is k, we associate it with a K-vector y, whose entries are $y_k = +1$ and $y_j = -1$ for $j \neq k$. (We can write this vector as $y = 2e_k - \mathbf{1}$.) Define $\tilde{y} = (\tilde{f}_1(x), \ldots, \tilde{f}_K(x))$, which is our (real-valued or continuous) prediction of the label y. Our multi-class prediction is given by $\hat{f}(x) = \operatorname{argmax}_{k=1,\ldots,K} \tilde{f}_k(x)$. Show that $\hat{f}(x)$ is also the index of the nearest neighbor of $\tilde{y}$ among the vectors $2e_k - \mathbf{1}$, for $k = 1, \ldots, K$. In other words, our guess $\hat{y}$ for the class is the nearest neighbor of our continuous prediction $\tilde{y}$, among the vectors that encode the class labels.

Solution. Let $\tilde{y}$ be any K-vector. Its distance squared to $2e_k - \mathbf{1}$, which represents class k, is

$$\begin{aligned} \|\tilde{y} - (2e_k - \mathbf{1})\|^2 &= \|(\tilde{y} + \mathbf{1}) - 2e_k\|^2 \\ &= \|\tilde{y} + \mathbf{1}\|^2 - 4(\tilde{y} + \mathbf{1})^T e_k + 4 \\ &= \|\tilde{y} + \mathbf{1}\|^2 - 4\tilde{y}_k. \end{aligned}$$

The first term does not depend on k. To minimize the second term (*i.e.*, to find the nearest neighbor), we should choose k so that $\tilde{y}_k$ is largest, *i.e.*, we should choose index $\operatorname{argmax}_{k=1,\ldots,K} \tilde{y}_k$.

14.10 *One-versus-one multi-class classifier.* In §14.3.1 we construct a K-class classifier from K Boolean classifiers that attempt to distinguish each class from the others. In this exercise we describe another method for constructing a K-class classifier. We first develop a Boolean classifier for every *pair* of classes i and j, $i < j$. There are $K(K-1)/2$ such pairs of classifiers, called *one-versus-one* classifiers. Given a feature vector x, we let $\hat{y}_{ij}$ be the prediction of the i-versus-j classifier, with $\hat{y}_{ij} = 1$ meaning that the one-versus-one classifier is guessing that $y = i$. We consider $\hat{y}_{ij} = 1$ as one 'vote' for class i, and $\hat{y}_{ij} = -1$ as one 'vote' for class j. We obtain the final estimated class by *majority voting*: We take $\hat{y}$ as the class that has the most votes. (We can break ties in some simple way, like taking the smallest index that achieves the largest number of votes.)

(a) Construct the least squares classifier, and the one-versus-one classifier, for a multi-class (training) data set. Find the confusion matrices, and the error rates, of the two classifiers on both the training data set and a separate test data set.

(b) Compare the complexity of computing the one-versus-one multi-class classifier with the complexity of the least squares multi-class classifier (see page 300). Assume the training set contains N/K examples of each class and that N/K is much greater than the number of features p. Distinguish two methods for the one-versus-one multi-class classifier. The first, naïve, method solves $K(K-1)/2$ least squares problem with N/K rows and p columns. The second, more efficient, method precomputes the Gram matrices $G_i = A_i A_i^T$ for $i = 1, \ldots, K$, where the rows of the $(N/K) \times p$ matrix A_i are the training example for class i, and uses the pre-computed Gram matrices to speed up the solution of the $K(K-1)/2$ least squares problems.

Solution.

(a) We use the MNIST data set from the chapter, *i.e.*, ten classes of images, each represented by a vector of length 493. The error rate on the training set is 6.5%. The confusion matrix is as follows.

	Prediction										
Digit	0	1	2	3	4	5	6	7	8	9	Total
0	5799	2	17	8	14	21	23	6	32	1	5923
1	2	6623	37	18	4	15	2	11	22	8	6742
2	50	79	5488	52	67	28	44	47	86	17	5958
3	33	44	120	5547	7	176	22	49	92	41	6131
4	14	20	15	2	5571	10	16	20	6	168	5842
5	45	43	41	140	26	4964	92	8	45	17	5421
6	26	16	43	2	36	86	5680	0	28	1	5918
7	9	81	55	5	82	12	1	5860	6	154	6265
8	31	198	47	117	43	144	38	31	5134	68	5851
9	22	13	21	82	164	35	2	153	30	5427	5949
All	6031	7119	5884	5973	6014	5491	5920	6185	5481	5902	60000

These results are much better than the error rate of 14.5% and the confusion matrix in table 14.11 for the classifier discussed at the beginning of section 14.3.3.

The error on the test set is 7.1%, with the following confusion matrix. (Compare with the test error of 13.9% and the confusion matrix 14.11. for the classifier of section 14.3.3.)

	Prediction										
Digit	0	1	2	3	4	5	6	7	8	9	Total
0	961	0	3	1	1	4	6	3	1	0	980
1	0	1122	3	3	1	1	2	1	2	0	1135
2	9	22	930	13	13	5	11	7	22	0	1032
3	8	1	16	926	3	20	3	10	18	5	1010
4	2	2	7	1	934	2	6	3	2	23	982
5	12	6	2	31	9	799	14	2	13	4	892
6	8	5	11	0	8	20	904	0	2	0	958
7	1	17	17	3	10	1	0	955	2	22	1028
8	8	16	10	20	11	34	12	10	840	13	974
9	6	5	1	10	31	11	0	23	5	917	1009
All	1015	1196	1000	1008	1021	897	958	1014	907	984	10000

(b) The least squares multi-class classifier has complexity $2Np^2 + Kp^2 \approx 2Np^2$, as discussed on page 300.

The naïve method for the one-versus-one classifier solves $K(K-1)/2$ least squares problems of size $(N/K) \times p$. The complexity is

$$\frac{K(K-1)}{2}\frac{2Np^2}{K} \approx KNp^2.$$

This is K times more expensive than the least squares method.

The improved method first computes K Gram matrices $G_i = A_i A_i^T$. The complexity of each is $(N/K)p^2$, so the total complexity for the K Gram matrices is Np^2. Given the Gram matrices, we can compute the Boolean classifier for classes i and j by solving the equation

$$(A_i^T A_i + A_j^T A_j)\beta = (A_i^T \mathbf{1} - A_j^T \mathbf{1}).$$

The coefficient matrix is $G_i + G_j$. This addition requires p^2 flops, and solving the equation takes $2p^3$ flops. The total complexity for the $K(K-1)/2$ least squares problems is roughly $(K^2/2)(2p^3) = K^2p^3$ flops. Adding this to to cost of computing the Gram matrices gives a total complexity

$$Np^2 + K^2p^3.$$

For small K this is dominated by Np^2, so we obtain roughly the same complexity as the least squares multi-class classifier.

14.11 *Equalizer design from training message.* We consider an electronic communication system, with message to be sent given by an N-vector s, whose entries are -1 or 1, and received signal y, where $y = c*s$, where c is an n-vector, the channel impulse response. The receiver applies equalization to the received signal, which means that it computes $\tilde{y} = h*y = h*c*s$, where h is an n-vector, the equalizer impulse response. The receiver then estimates the original message using $\hat{s} = \mathbf{sign}(\tilde{y}_{1:N})$. This works well if $h * c \approx e_1$. (See exercise 7.15.) If the channel impulse response c is known or can be measured, we can design or choose h using least squares, as in exercise 12.6.

In this exercise we explore a method for choosing h directly, without estimating or measuring c. The sender first sends a message that is *known* to the receiver, called the *training message*, s^{train}. (From the point of view of communications, this is wasted transmission, and is called *overhead*.) The receiver receives the signal $y^{\text{train}} = c * s^{\text{train}}$ from the training message, and then chooses h to minimize $\|(h * y^{\text{train}})_{1:N} - s^{\text{train}}\|^2$. (In practice, this equalizer is used until the bit error rate increases, which means the channel has changed, at which point another training message is sent.) Explain how this method is the same as least squares classification. What are the training data $x^{(i)}$ and $y^{(i)}$? What is the least squares problem that must be solved to determine the equalizer impulse response h?

Solution.

Chapter 15

Multi-objective least squares

Exercises

15.1 *A scalar multi-objective least squares problem.* We consider the special case of the multi-objective least squares problem in which the variable x is a scalar, and the k matrices A_i are all 1×1 matrices with value $A_i = 1$, so $J_i = (x-b_i)^2$. In this case our goal is to choose a number x that is simultaneously close to all the numbers $b_1, \ldots, b_k$. Let $\lambda_1, \ldots, \lambda_k$ be positive weights, and $\hat{x}$ the minimizer of the weighted objective (15.1). Show that $\hat{x}$ is a weighted average (or convex combination; see page 17) of the numbers $b_1, \ldots, b_k$, *i.e.*, it has the form

$$x = w_1 b_1 + \cdots + w_k b_k,$$

where w_i are nonnegative and sum to one. Give an explicit formula for the combination weights w_i in terms of the multi-objective least squares weights λ_i.

Solution. The problem is to minimize

$$\lambda_1(x-b_1)^2 + \lambda_2(x-b_2)^2 + \cdots + \lambda_k(x-b_k)^2$$

over the scalar x. If we set the derivative to zero we get a simple equation

$$2\lambda_1(x-b_1) + \cdots + 2\lambda_k(x-b_k) = 0$$

with solution

$$\hat{x} = \frac{\lambda_1 b_1 + \cdots + \lambda_k b_k}{\lambda_1 + \cdots + \lambda_k}.$$

The same result follows from the general formula (15.3) with $A_i = 1$ (*i.e.*, the 1×1 matrix).

The formula for $\hat{x}$ has the required form with

$$w_i = \frac{\lambda_i}{\lambda_1 + \cdots + \lambda_k}, \quad i = 1, \ldots, k.$$

The combination weights are proportional to the multi-objective least squares weights.

15.2 Consider the regularized data fitting problem (15.7). Recall that the elements in the first column of A are one. Let $\hat{\theta}$ be the solution of (15.7), *i.e.*, the minimizer of

$$\|A\theta - y\|^2 + \lambda(\theta_2^2 + \cdots + \theta_p^2),$$

and let $\tilde{\theta}$ be the minimizer of

$$\|A\theta - y\|^2 + \lambda\|\theta\|^2 = \|A\theta - y\|^2 + \lambda(\theta_1^2 + \theta_2^2 + \cdots + \theta_p^2),$$

in which we also penalize θ_1. Suppose columns 2 through p of A have mean zero (for example, because features $2, \ldots, p$ have been standardized on the data set; see page 269). Show that $\hat{\theta}_k = \tilde{\theta}_k$ for $k = 2, \ldots, p$.

Solution. Suppose A has N rows and define $B = A_{1:N,2:p}$. We have $\mathbf{1}^T B = 0$ because each column of B has mean zero.

The first multi-objective problem is equivalent to the least squares problem of minimizing

$$\left\| \begin{bmatrix} \mathbf{1} & B \\ 0 & \sqrt{\lambda} I \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_{2:p} \end{bmatrix} - \begin{bmatrix} y \\ 0 \end{bmatrix} \right\|^2.$$

The normal equations are

$$\begin{bmatrix} \mathbf{1} & B \\ 0 & \sqrt{\lambda} I \end{bmatrix}^T \begin{bmatrix} \mathbf{1} & B \\ 0 & \sqrt{\lambda} I \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_{2:p} \end{bmatrix} = \begin{bmatrix} \mathbf{1} & B \\ 0 & \sqrt{\lambda} I \end{bmatrix}^T \begin{bmatrix} y \\ 0 \end{bmatrix},$$

which reduces to

$$\begin{bmatrix} N & \mathbf{1}^T B \\ B^T\mathbf{1} & B^TB + \lambda I \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_{2:p} \end{bmatrix} = \begin{bmatrix} \mathbf{1}^T y \\ B^T y \end{bmatrix}.$$

Since $\mathbf{1}^T B = 0$, the solution is

$$\hat{\theta}_1 = \frac{1}{N}\mathbf{1}^T y, \qquad \hat{\theta}_{2:p} = (B^TB + \lambda I)^{-1}B^T y.$$

The second multi-objective problem is equivalent to the least squares problem of minimizing

$$\left\| \begin{bmatrix} \mathbf{1} & B \\ \sqrt{\lambda} & 0 \\ 0 & \sqrt{\lambda}I \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_{2:p} \end{bmatrix} - \begin{bmatrix} y \\ 0 \\ 0 \end{bmatrix} \right\|^2.$$

The normal equations are

$$\begin{bmatrix} \mathbf{1} & B \\ \sqrt{\lambda} & 0 \\ 0 & \sqrt{\lambda}I \end{bmatrix}^T \begin{bmatrix} \mathbf{1} & B \\ \sqrt{\lambda} & 0 \\ 0 & \sqrt{\lambda}I \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_{2:p} \end{bmatrix} = \begin{bmatrix} \mathbf{1} & B \\ \sqrt{\lambda} & 0 \\ 0 & \sqrt{\lambda}I \end{bmatrix}^T \begin{bmatrix} y \\ 0 \\ 0 \end{bmatrix},$$

which reduces to

$$\begin{bmatrix} N+\lambda & \mathbf{1}^T B \\ B^T\mathbf{1} & B^TB + \lambda I \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_{2:p} \end{bmatrix} = \begin{bmatrix} \mathbf{1}^T y \\ B^T y \end{bmatrix}.$$

Since $\mathbf{1}^T B = 0$, the solution is

$$\tilde{\theta}_1 = \frac{1}{N+\lambda}\mathbf{1}^T y, \qquad \tilde{\theta}_{2:p} = (B^TB + \lambda I)^{-1}B^T y.$$

We see that $\hat{\theta}_{2:p} = \tilde{\theta}_{2:p}$.

15.3 *Weighted Gram matrix.* Consider a multi-objective least squares problems with matrices $A_1, \ldots, A_k$ and positive weights $\lambda_1, \ldots, \lambda_k$. The matrix

$$G = \lambda_1 A_1^T A_1 + \cdots + \lambda_k A_k^T A_k$$

is called the *weighted Gram matrix*; it is the Gram matrix of the stacked matrix $\tilde{A}$ (given in (15.2)) associated with the multi-objective problem. Show that G is invertible provided there is no nonzero vector x that satisfies $A_1 x = 0, \ldots, A_k x = 0$.

Solution. Suppose $Gx = 0$. Then

$$\begin{aligned} 0 &= x^T G x \\ &= \lambda_1 x^T A_1^T A_1 x + \cdots + \lambda_k x^T A_k^T A_k x \\ &= \lambda_1 \|A_1 x\|^2 + \cdots + \lambda_k \|A_k x\|^2. \end{aligned}$$

This is only possible if $A_1 x = 0, \ldots, A_k x = 0$. By our assumption, the only vector that satisfies this is $x = 0$. Hence $Gx = 0$ holds only if $x = 0$. Therefore G is invertible.

15.4 *Robust approximate solution of linear equations.* We wish to solve the square set of n linear equations $Ax = b$ for the n-vector x. If A is invertible the solution is $x = A^{-1}b$. In this exercise we address an issue that comes up frequently: We don't know A exactly. One simple method is to just choose a typical value of A and use it. Another method, which we explore here, takes into account the variation in the matrix A. We find a set of K versions of A, and denote them as $A^{(1)}, \ldots, A^{(K)}$. (These could be found by measuring the matrix A at different times, for example.) Then we choose x so as to minimize

$$\|A^{(1)}x - b\|^2 + \cdots + \|A^{(K)}x - b\|^2,$$

the sum of the squares of residuals obtained with the K versions of A. This choice of x, which we denote $x^{\rm rob}$, is called a *robust* (approximate) solution. Give a formula for $x^{\rm rob}$, in terms of $A^{(1)}, \ldots, A^{(K)}$ and b. (You can assume that a matrix you construct has linearly independent columns.) Verify that for $K = 1$ your formula reduces to $x^{\rm rob} = (A^{(1)})^{-1}b$.

Solution. The problem is equivalent to minimizing

$$\left\| \begin{bmatrix} A^{(1)} \\ A^{(2)} \\ \vdots \\ A^{(K)} \end{bmatrix} x - \begin{bmatrix} b^{(1)} \\ b^{(2)} \\ \vdots \\ b^{(K)} \end{bmatrix} \right\|^2.$$

This is a least squares problem: we minimize $\|\tilde{A}x - \tilde{b}\|^2$ with

$$\tilde{A} = \begin{bmatrix} A^{(1)} \\ A^{(2)} \\ \vdots \\ A^{(K)} \end{bmatrix}, \qquad \tilde{b} = \begin{bmatrix} b^{(1)} \\ b^{(2)} \\ \vdots \\ b^{(K)} \end{bmatrix}.$$

The solution is

$$\hat{x} = (\tilde{A}^T\tilde{A})^{-1}\tilde{A}^T\tilde{b} = \left(\sum_{k=1}^{K}(A^{(k)})^TA^{(k)}\right)^{-1}\left(\sum_{k=1}^{K}(A^{(k)})^Tb^{(k)}\right).$$

15.5 *Some properties of bi-objective least squares.* Consider the bi-objective least squares problem with objectives

$$J_1(x) = \|A_1x - b_1\|^2, \qquad J_2(x) = \|A_2x - b_2\|^2.$$

For $\lambda > 0$, let $\hat{x}(\lambda)$ denote the minimizer of $J_1(x) + \lambda J_2(x)$. (We assume the columns of the stacked matrix are linearly independent.) We define $J_1^\star(\lambda) = J_1(\hat{x}(\lambda))$ and $J_2^\star(\lambda) = J_2(\hat{x}(\lambda))$, the values of the two objectives as functions of the weight parameter. The optimal trade-off curve is the set of points $(J_1^\star(\lambda), J_2^\star(\lambda))$, as λ varies over all positive numbers.

(a) *Bi-objective problems without trade-off.* Suppose μ and γ are different positive weights, and $\hat{x}(\mu) = \hat{x}(\gamma)$. Show that $\hat{x}(\lambda)$ is constant for all $\lambda > 0$. Therefore the point $(J_1^\star(\lambda), J_2^\star(\lambda))$ is the same for all λ and the trade-off curve collapses to a single point.

(b) *Effect of weight on objectives in a bi-objective problem.* Suppose $\hat{x}(\lambda)$ is not constant. Show the following: for $\lambda < \mu$, we have

$$J_1^\star(\lambda) < J_1^\star(\mu), \qquad J_2^\star(\lambda) > J_2^\star(\mu).$$

This means that if you increase the weight (on the second objective), the second objective goes down, and the first objective goes up. In other words the trade-off curve slopes downward.

Hint. Resist the urge to write out any equations or formulas. Use the fact that $\hat{x}(\lambda)$ is the unique minimizer of $J_1(x) + \lambda J_2(x)$, and similarly for $\hat{x}(\mu)$, to deduce the inequalities

$$J_1^\star(\mu) + \lambda J_2^\star(\mu) > J_1^\star(\lambda) + \lambda J_2^\star(\lambda), \qquad J_1^\star(\lambda) + \mu J_2^\star(\lambda) > J_1^\star(\mu) + \mu J_2^\star(\mu).$$

Combine these inequalities to show that $J_1^\star(\lambda) < J_1^\star(\mu)$ and $J_2^\star(\lambda) > J_2^\star(\mu)$.

(c) *Slope of the trade-off curve.* The slope of the trade-off curve at the point $(J_1^\star(\lambda), J_2^\star(\lambda))$ is given by

$$S = \lim_{\mu \to \lambda} \frac{J_2^\star(\mu) - J_2^\star(\lambda)}{J_1^\star(\mu) - J_1^\star(\lambda)}.$$

(This limit is the same if μ approaches λ from below or from above.) Show that $S = -1/\lambda$. This gives another interpretation of the parameter λ: $(J_1^\star(\lambda), J_2^\star(\lambda))$ is the point on the trade-off curve where the curve has slope $-1/\lambda$.

Hint. First assume that μ approaches λ from above (meaning, $\mu > \lambda$) and use the inequalities in the hint for part (b) to show that $S \geq -1/\lambda$. Then assume that μ approaches λ from below and show that $S \leq -1/\lambda$.

Solution.

(a) Define $\tilde{x} = \hat{x}(\mu) = \hat{x}(\gamma)$. This point satisfies

$$(A_1^T A_1 + \mu A_2^T A_2)\tilde{x} = A_1^T b_1 + \mu A_2^T b_2, \qquad (A_1^T A_1 + \gamma A_2^T A_2)\tilde{x} = A_1^T b_1 + \gamma A_2^T b_2.$$

Consider any positive λ different from μ and γ. Then

$$\begin{aligned}(A_1^T A_1 + \lambda A_2^T A_2)\tilde{x} &= \frac{\lambda - \gamma}{\mu - \gamma}(A_1^T A_1 + \mu A_2^T A_2)\tilde{x} + \frac{\mu - \lambda}{\mu - \gamma}(A_1^T A_1 + \gamma A_2^T A_2)\tilde{x} \\ &= \frac{\lambda - \gamma}{\mu - \gamma}(A_1^T b_1 + \mu A_2^T b_2) + \frac{\mu - \lambda}{\mu - \gamma}(A_1^T b_1 + \gamma A_2^T b_2) \\ &= A_1^T b_1 + \lambda A_2^T b_2.\end{aligned}$$

This shows that $\hat{x}(\lambda) = \tilde{x}$.

(b) We follow the hint. Since $\hat{x}(\lambda)$ minimizes $J_1(x) + \lambda J_2(x)$, and it is the unique minimizer by the assumption of linear independence of the stacked matrix, we have

$$J_1(x) + \lambda J_2(x) > J_1(\hat{x}(\lambda)) + \lambda J_2(\hat{x}(\lambda))$$

for any $x \neq \hat{x}(\lambda)$. In particular, this holds for $x = \hat{x}(\mu)$, which gives us the inequality

$$J_1^\star(\mu) + \lambda J_2^\star(\mu) > J_1^\star(\lambda) + \lambda J_2^\star(\lambda).$$

Reversing the roles of λ and μ, we get the other inequality in the hint,

$$J_1^\star(\lambda) + \mu J_2^\star(\lambda) > J_1^\star(\mu) + \mu J_2^\star(\mu).$$

We add these two inequalities to get

$$J_1^\star(\mu) + \lambda J_2^\star(\mu) + J_1^\star(\lambda) + \mu J_2^\star(\lambda) > J_1^\star(\lambda) + \lambda J_2^\star(\lambda) + J_1^\star(\mu) + \mu J_2^\star(\mu).$$

Simplifying this gives

$$(\lambda - \mu)(J_2^\star(\mu) - J_2^\star(\lambda)) > 0.$$

This immediately shows that when $\mu > \lambda$, we have $J_2^\star(\mu) < J_2^\star(\lambda)$.

To get the other inequality we re-arrange the first inequality in the hint to get

$$J_1^\star(\mu) - J_1^\star(\lambda) > \lambda(J_2^\star(\lambda) - J_2^\star(\mu)).$$

When $\mu > \lambda$, Since the right-hand side is nonnegative when $\mu > \lambda$, we have $J_1^\star(\mu) > J_1^\star(\lambda)$.

(c) We start from the inequalities mentioned in the hint. The first inequality is

$$J_1^\star(\mu) - J_1^\star(\lambda) > -\lambda(J_2^\star(\mu) - J_2^\star(\lambda))$$

and holds for any positive, distinct values of λ and μ. Assume that $\mu > \lambda$. By part (b), we have $J_1^\star(\lambda) < J_1^\star(\mu)$, so dividing the inequality above by $-\lambda(J_1^\star(\mu) - J_1^\star(\lambda))$ (which is negative) we get

$$-1/\lambda < \frac{J_2^\star(\mu) - J_2^\star(\lambda)}{J_1^\star(\mu) - J_1^\star(\lambda)}.$$

Letting $\mu \to \lambda$ we get $-1/\lambda \leq S$.

Now assume that $\mu < \lambda$. Reversing the roles of λ and μ, we find that $-1/\lambda \geq S$.

15.6 *Least squares with smoothness regularization.* Consider the weighted sum least squares objective

$$\|Ax - b\|^2 + \lambda\|Dx\|^2,$$

where the n-vector x is the variable, A is an $m \times n$ matrix, D is the $(n-1) \times n$ difference matrix, with ith row $(e_{i+1} - e_i)^T$, and $\lambda > 0$. Although it does not matter in this problem, this objective is what we would minimize if we want an x that satisfies $Ax \approx b$, and has entries that are smoothly varying. We can express this objective as a standard least squares objective with a stacked matrix of size $(m + n - 1) \times n$.

Show that the stacked matrix has linearly independent columns if and only if $A\mathbf{1} \neq 0$, *i.e.*, the sum of the columns of A is not zero.

Solution. The stacked matrix is

$$S = \begin{bmatrix} A \\ \sqrt{\lambda}D \end{bmatrix}.$$

If $A\mathbf{1} = 0$, then $S\mathbf{1} = 0$ so the stacked matrix does not have linearly independent columns. So the condition is necessary.

To show it is sufficient, assume $A\mathbf{1} \neq 0$. Suppose that $Sx = 0$. This is the same as $Ax = 0$, $\sqrt{\lambda}Dx = 0$. Since $\lambda > 0$, the second condition reduces to $Dx = 0$. But this is only possible if x is a constant vector, *i.e.*, $x = \alpha\mathbf{1}$ for some scalar α. From the first equation $A(\alpha\mathbf{1}) = 0$, and since $A\mathbf{1} \neq 0$, this means $\alpha = 0$. Therefore $Sx = 0$ only if $x = 0$, so S has linearly independent columns.

15.7 *Greedy regulation policy.* Consider a linear dynamical system given by $x_{t+1} = Ax_t + Bu_t$, where the n-vector x_t is the state at time t, and the m-vector u_t is the input at time t. The goal in regulation is to choose the input so as to make the state small. (In applications, the state $x_t = 0$ corresponds to the desired operating point, so small x_t means the state is close to the desired operating point.) One way to achieve this goal is to choose u_t so as to minimize

$$\|x_{t+1}\|^2 + \rho\|u_t\|^2,$$

where ρ is a (given) positive parameter that trades off using a small input versus making the (next) state small. Show that choosing u_t this way leads to a state feedback policy $u_t = Kx_t$, where K is an $m \times n$ matrix. Give a formula for K (in terms of A, B, and ρ). If an inverse appears in your formula, state the conditions under which the inverse exists.

Remark. This policy is called *greedy* or *myopic* since it does not take into account the effect of the input u_t on future states, beyond x_{t+1}. It can work very poorly in practice.

Solution. The input u_t is chosen to minimize

$$\|Ax_t + Bu_t\|^2 + \rho\|u_t\|^2 = \left\| \begin{bmatrix} B \\ \sqrt{\rho}I \end{bmatrix} u_t - \begin{bmatrix} -Ax_t \\ 0 \end{bmatrix} \right\|^2.$$

The solution is

$$\begin{aligned} u_t &= \left(\begin{bmatrix} B \\ \sqrt{\rho} I \end{bmatrix}^T \begin{bmatrix} B \\ \sqrt{\rho} I \end{bmatrix} \right)^{-1} \left(\begin{bmatrix} B \\ \sqrt{\rho} I \end{bmatrix}^T \begin{bmatrix} -Ax_t \\ 0 \end{bmatrix} \right) \\ &= -\left(B^T B + \rho I\right)^{-1} B^T A x_t. \end{aligned}$$

This is a state feedback policy $u_t = Kx_t$ with $K = -\left(B^T B + \rho I\right)^{-1} B^T A$. The inverse in this expression exists, since the columns of the stacked matrix are linearly independent.

15.8 *Estimating the elasticity matrix.* In this problem you create a standard model of how demand varies with the prices of a set of products, based on some observed data. There are n different products, with (positive) prices given by the n-vector p. The prices are held constant over some period, say, a day. The (positive) demands for the products over the day are given by the n-vector d. The demand in any particular day varies, but it is thought to be (approximately) a function of the prices.

The *nominal prices* are given by the n-vector p^{nom}. You can think of these as the prices that have been charged in the past for the products. The *nominal demand* is the n-vector d^{nom}. This is the average value of the demand, when the prices are set to p^{nom}. (The actual daily demand fluctuates around the value d^{nom}.) You know both p^{nom} and d^{nom}. We will describe the prices by their (fractional) variations from the nominal values, and the same for demands. We define δ^p and δ^d as the (vectors of) relative price change and demand change:

$$\delta_i^p = \frac{p_i - p_i^{\text{nom}}}{p_i^{\text{nom}}}, \quad \delta_i^d = \frac{d_i - d_i^{\text{nom}}}{d_i^{\text{nom}}}, \quad i = 1, \ldots, n.$$

So $\delta_3^p = +0.05$ means that the price for product 3 has been increased by 5% over its nominal value, and $\delta_5^d = -0.04$ means that the demand for product 5 in some day is 4% below its nominal value.

Your task is to build a model of the demand as a function of the price, of the form

$$\delta^d \approx E\delta^p,$$

where E is the $n \times n$ elasticity matrix. You don't know E, but you do have the results of some experiments in which the prices were changed a bit from their nominal values for one day, and the day's demands were recorded. This data has the form

$$(p_1, d_1), \ldots, (p_N, d_N),$$

where p_i is the price for day i, and d_i is the observed demand.

Explain how you would estimate E, given this price-demand data. Be sure to explain how you will test for, and (if needed) avoid over-fit. *Hint.* Formulate the problem as a matrix least squares problem; see page 233.

Remark. Note the difference between elasticity estimation and demand shaping, discussed on page 315. In demand shaping, we know the elasticity matrix and are choosing prices; in elasticity estimation, we are guessing the elasticity matrix from some observed price and demand data.

Solution. Let Δ^p denote the $n \times N$ matrix whose columns are the price change vectors δ_i^p, and let Δ^d denote the $n \times N$ matrix whose columns are the corresponding demand vectors δ_i^d. Our job is to find an $n \times n$ matrix E so that $\Delta^d \approx E\Delta^p$. This can be achieved by minimizing

$$\|E\Delta^p - \Delta^d\|^2 = \|(\Delta^p)^T E^T - (\Delta^d)^T\|^2.$$

This is a matrix least squares problem: minimize $\|AX - B\|^2$ with $A = (\Delta^p)^T$, $B = (\Delta^d)^T$, and variable $X = E^T$.

We can check our model by out-of-sample validation. To avoid over-fit, we might need to add regularization, *i.e.*, add terms $\lambda_i \|e_i\|^2$ to the objective above, where $\lambda_i > 0$ is a parameter and e_i^T is the ith row of E.

15.9 *Regularizing stratified models.* In a *stratified* model (see page 272), we divide the data into different sets, depending on the value of some (often Boolean) feature, and then fit a separate model for each of these two data sets, using the remaining features. As an example, to develop a model of some health outcome we might build a separate model for women and for men. In some cases better models are obtained when we encourage the different models in a stratified model to be close to each other. For the case of stratifying on one Boolean feature, this is done by choosing the two model parameters $\theta^{(1)}$ and $\theta^{(2)}$ to minimize

$$\|A^{(1)}\theta^{(1)} - y^{(1)}\|^2 + \|A^{(2)}\theta^{(2)} - y^{(2)}\|^2 + \lambda\|\theta^{(1)} - \theta^{(2)}\|^2,$$

where $\lambda \geq 0$ is a parameter. The first term is the least squares residual for the first model on the first data set (say, women); the second term is the least squares residual for the second model on the second data set (say, men); the third term is a regularization term that encourages the two model parameters to be close to each other. Note that when $\lambda = 0$, we simply fit each model separately; when λ is very large, we are basically fitting one model to all the data. Of course the choice of an appropriate value of λ is obtained using out-of-sample validation (or cross-validation).

(a) Give a formula for the optimal $(\hat{\theta}^{(1)}, \hat{\theta}^{(2)})$. (If your formula requires one or more matrices to have linearly independent columns, say so.)

(b) *Stratifying across age groups.* Suppose we fit a model with each data point representing a person, and we stratify over the person's *age group*, which is a range of consecutive ages such as 18–24, 24–32, 33–45, and so on. Our goal is to fit a model for each age of k groups, with the parameters for adjacent age groups similar, or not too far, from each other. Suggest a method for doing this.

Solution.

(a) The problem is equivalent to minimizing $\|A\theta - b\|^2$ where

$$A = \begin{bmatrix} A^{(1)} & 0 \\ 0 & A^{(2)} \\ \sqrt{\lambda}I & -\sqrt{\lambda}I \end{bmatrix}, \qquad \theta = \begin{bmatrix} \theta^{(1)} \\ \theta^{(2)} \end{bmatrix} \qquad b = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ 0 \end{bmatrix}.$$

The solution is

$$\begin{aligned} \hat{\theta} &= (A^TA)^{-1}A^Tb \\ &= \begin{bmatrix} (A^{(1)})^TA^{(1)} + \lambda I & -\lambda I \\ -\lambda I & (A^{(2)})^TA^{(2)} + \lambda I \end{bmatrix}^{-1} \begin{bmatrix} (A^{(1)})^Ty^{(1)} \\ (A^{(2)})^Ty^{(2)} \end{bmatrix}. \end{aligned}$$

The inverse in this formula exists if the stacked matrix A has linearly independent columns. To see what this means for $A^{(1)}$ and $A^{(2)}$, suppose $x = (x_1, x_2)$ satisfies $Ax = 0$:

$$Ax = \begin{bmatrix} A^{(1)} & 0 \\ 0 & A^{(2)} \\ \sqrt{\lambda}I & -\sqrt{\lambda}I \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} A^{(1)}x_1 \\ A^{(2)}x_2 \\ \sqrt{\lambda}(x_1 - x_2) \end{bmatrix} = 0.$$

From the third block, $x_1 = x_2$. The other two blocks give the conditions $A^{(1)}x_1 = 0$ and $A^{(2)}x_2 = 0$. Since $x_1 = x_2$, this implies that $x_1 = 0$, $x_2 = 0$ if and only if the stacked matrix

$$\begin{bmatrix} A^{(1)} \\ A^{(2)} \end{bmatrix}$$

has linearly independent columns.

(b) We minimize

$$\|A^{(1)}\theta^{(1)} - y^{(1)}\|^2 + \cdots + \|A^{(k)}\theta^{(k)} - y^{(k)}\|^2 + \lambda\|\theta^{(1)} - \theta^{(2)}\|^2 + \lambda\|\theta^{(2)} - \theta^{(3)}\|^2 + \cdots + \lambda\|\theta^{(k-1)} - \theta^{(k)}\|^2.$$

15.10 *Estimating a periodic time series.* (See §15.3.2.) Suppose that the T-vector y is a measured time series, and we wish to approximate it with a P-periodic T-vector. For simplicity, we assume that $T = KP$, where K is an integer. Let $\hat{y}$ be the simple least squares fit, with no regularization, *i.e.*, the P-periodic vector that minimizes $\|\hat{y} - y\|^2$. Show that for $i = 1, \ldots, P-1$, we have

$$\hat{y}_i = \frac{1}{K}\sum_{k=1}^{K} y_{i+(k-1)P}.$$

In other words, each entry of the periodic estimate is the average of the entries of the original vector over the corresponding indices.

Solution. We let x be a P-vector corresponding to one cycle of $\hat{y}$, so $y = Ax$, where A is a matrix consisting of K copies of $I_{k\times K}$ stacked on top of each other. Then

$$A^TA = KI, \qquad A^Ty = \sum_{k=1}^{K} y_{((k-1)P+1):(kP)}.$$

Therefore the least squares estimate is

$$\hat{y} = (A^TA)^{-1}A^Ty = \frac{1}{K}\sum_{k=1}^{K} y_{((k-1)P+1):(kP)}.$$

In other words

$$\hat{y}_i = \frac{1}{K}\sum_{k=1}^{K} y_{(k-1)P+i}.$$

15.11 *General pseudo-inverse.* In chapter 11 we encountered the pseudo-inverse of a tall matrix with linearly independent columns, a wide matrix with linearly independent rows, and a square invertible matrix. In this exercise we describe the pseudo-inverse of a general matrix, *i.e.*, one that does not fit these categories. The general pseudo-inverse can be defined in terms of Tikhonov regularized inversion (see page 317). Let A be any matrix, and $\lambda > 0$. The Tikhonov regularized approximate solution of $Ax = b$, *i.e.*, unique minimizer of $\|Ax - b\|^2 + \lambda\|x\|^2$, is given by $(A^TA + \lambda I)^{-1}A^Tb$. The pseudo-inverse of A is defined as

$$A^\dagger = \lim_{\lambda\to 0}(A^TA + \lambda I)^{-1}A^T.$$

In other words, $A^\dagger b$ is the limit of the Tikhonov-regularized approximate solution of $Ax = b$, as the regularization parameter converges to zero. (It can be shown that this limit always exists.) Using the kernel trick identity (15.10), we can also express the pseudo-inverse as

$$A^\dagger = \lim_{\lambda\to 0} A^T(AA^T + \lambda I)^{-1}.$$

(a) What is the pseudo-inverse of the $m \times n$ zero matrix?

(b) Suppose A has linearly independent columns. Explain why the limits above reduce to our previous definition, $A^\dagger = (A^TA)^{-1}A^T$.

(c) Suppose A has linearly independent rows. Explain why the limits above reduce to our previous definition, $A^\dagger = A^T(AA^T)^{-1}$.

Hint. For parts (b) and (c), you can use the fact that the matrix inverse is a continuous function, which means that the limit of the inverse of a matrix is the inverse of the limit, provided the limit matrix is invertible.

Solution.

(a) We have $0^\dagger_{m\times n} = 0_{n\times m}$. To see this, we use

$$0^\dagger = \lim_{\lambda\to 0}(0^T 0 + \lambda I)^{-1}0^T = \lim_{\lambda\to 0} 0^T = 0^T.$$

(b) If A has linearly independent columns, A^TA is invertible, and we have

$$\lim_{\lambda\to 0}(A^TA + \lambda I)^{-1} = (A^TA)^{-1},$$

since $\lim_{\lambda\to 0}(A^TA+\lambda I) = A^TA$, which is invertible. It follows that the limits above reduce to $A^\dagger = (A^TA)^{-1}A^T$.

(c) If A has linearly independent rows, AA^T is invertible, and we use a similar argument using the second limit,

$$A^\dagger = \lim_{\lambda\to 0}(AA^T + \lambda I)^{-1} = (AA^T)^{-1}.$$

This recovers our previous formula $A^\dagger = A^T(AA^T)^{-1}$ for a matrix with linearly independent rows.

Chapter 16

Constrained least squares

Exercises

16.1 *Smallest right inverse.* Suppose the $m \times n$ matrix A is wide, with linearly independent rows. Its pseudo-inverse $A^\dagger$ is a right inverse of A. In fact, there are many right inverses of A and it turns out that $A^\dagger$ is the smallest one among them, as measured by the matrix norm. In other words, if X satisfies $AX = I$, then $\|X\| \geq \|A^\dagger\|$. You will show this in this problem.

(a) Suppose $AX = I$, and let $x_1, \ldots, x_m$ denote the columns of X. Let b_j denote the jth column of $A^\dagger$. Explain why $\|x_j\|^2 \geq \|b_j\|^2$. *Hint.* Show that $z = b_j$ is the vector of smallest norm that satisfies $Az = e_j$, for $j = 1, \ldots, m$.

(b) Use the inequalities from part (a) to establish $\|X\| \geq \|A^\dagger\|$.

Solution.

(a) Finding the z that minimizes $\|z\|^2$ subject to $Az = e_j$ is a least norm problem. The solution is $A^\dagger e_j = b_j$.

(b) Since X is a right inverse of A, we have $Ax_j = e_j$ for $j = 1, \ldots, m$. It follows from part (a) that $\|x_j\|^2 \geq \|b_j\|^2$ for $j = 1, \ldots, m$. Combining these inequalities, we conclude

$$\|X\|^2 = \sum_{j=1}^{m} \|x_j\|^2 \geq \sum_{j=1}^{m} \|b_j\|^2 = \|A^\dagger\|^2.$$

16.2 *Matrix least norm problem.* The matrix least norm problem is

$$\begin{array}{ll} \text{minimize} & \|X\|^2 \\ \text{subject to} & CX = D, \end{array}$$

where the variable to be chosen is the $n \times k$ matrix X; the $p \times n$ matrix C and the $p \times k$ matrix D are given. Show that the solution of this problem is $\hat{X} = C^\dagger D$, assuming the rows of C are linearly independent. *Hint.* Show that we can find the columns of X independently, by solving a least norm problem for each one.

Solution. The squared norm of a matrix is the sum of the squares of its column norms:

$$\|X\|^2 = \sum_{j=1}^{k} \|x_j\|^2,$$

where x_j denotes the jth column of X. The matrix equation $CX = D$ is equivalent to k vector equations $Cx_j = d_j$, for $j = 1, \ldots, k$, where d_j is column j of D. As a consequence of these two observations, the problem can be written as

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{j=1}^{k} \|x_j\|^2 \\ \text{subject to} & Cx_j = d_j, \quad j = 1, \ldots, k. \end{array}$$

This shows that each column can be chosen independently of the others. The optimal choice for x_j is the solution of

$$\begin{array}{ll} \text{minimize} & \|x_j\|^2 \\ \text{subject to} & Cx_j = d_j. \end{array}$$

This is a standard least norm problem, with solution $\hat{x}_j = C^\dagger d_j$. We conclude that

$$\hat{X} = \begin{bmatrix} C^\dagger d_1 & C^\dagger d_2 & \cdots & C^\dagger d_k \end{bmatrix} = C^\dagger D.$$

16.3 *Closest solution to a given point.* Suppose the wide matrix A has linearly independent rows. Find an expression for the point x that is closest to a given vector y (*i.e.*, minimizes $\|x-y\|^2$) among all vectors that satisfy $Ax=b$.

Remark. This problem comes up when x is some set of inputs to be found, $Ax=b$ represents some set of requirements, and y is some nominal value of the inputs. For example, when the inputs represent actions that are re-calculated each day (say, because b changes every day), y might be yesterday's action, and the today's action x found as above gives the least change from yesterday's action, subject to meeting today's requirements.

Solution. We rewrite the problem as a least-norm problem by making a change of variables $z=x-y$. The new variable z obeys the equality constraint

$$Az = A(x-y) = b - Ay.$$

The problem is now to minimize $\|z\|^2$ subject to $Az = b-Ay$. This is a least norm problem, with solution

$$z = A^\dagger(b-Ay).$$

Going back to the original variable x, we find

$$x = z + y = A^\dagger(b-Ay) + y = A^\dagger b + (I - A^\dagger A)y.$$

16.4 *Nearest vector with a given average.* Let a be an n-vector and β a scalar. How would you find the n-vector x that is closest to a among all n-vectors that have average value β? Give a formula for x and describe it in English.

Solution. $x = a + (\beta - \mathbf{avg}(a))\mathbf{1}$.

Let's justify this. We seek the point x that minimizes $\|x-a\|^2$ subject to $\mathbf{1}^T x = n\beta$. The optimality conditions are

$$\begin{bmatrix} 2I & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} = \begin{bmatrix} 2a \\ n\beta \end{bmatrix}.$$

From the first block, we get $x = a - (z/2)\mathbf{1}$. Substituting in the second equation $\mathbf{1}^T x = n\beta$ gives $\mathbf{1}^T a - nz/2 = n\beta$. Therefore $z = 2(\beta - \mathbf{1}^T a/n) = 2(\beta - \mathbf{avg}(a))$ and

$$x = a - (z/2)\mathbf{1} = a + (\beta - \mathbf{avg}(a))\mathbf{1}.$$

16.5 *Checking constrained least squares solution.* Generate a random 20×10 matrix A and a random 5×10 matrix C. Then generate random vectors b and d of appropriate dimensions for the constrained least squares problem

$$\begin{array}{ll} \text{minimize} & \|Ax-b\|^2 \\ \text{subject to} & Cx = d. \end{array}$$

Compute the solution $\hat{x}$ by forming and solving the KKT equations. Verify that the constraints very nearly hold, *i.e.*, $C\hat{x} - d$ is very small. Find the least norm solution x^{ln} of $Cx=d$. The vector x^{ln} also satisfies $Cx=d$ (very nearly). Verify that $\|Ax^{\text{ln}} - b\|^2 > \|A\hat{x} - b\|^2$.

16.6 *Modifying a diet to meet nutrient requirements.* (Continuation of exercise 8.9.) The current daily diet is specified by the n-vector d^{curr}. Explain how to find the closest diet d^{mod} to d^{curr} that satisfies the nutrient requirements given by the m-vector n^{des}, and has the same cost as the current diet d^{curr}.

Solution. We minimize $\|d - d^{\text{curr}}\|^2$ subject to $Nd = n^{\text{des}}$ and $c^T d = c^T d^{\text{curr}}$. If we use the variable $x = d - d^{\text{curr}}$, this is a least norm problem: Minimize $\|x\|^2$ subject to

$$\begin{bmatrix} N \\ c^T \end{bmatrix} x = \begin{bmatrix} n^{\text{des}} - Nd^{\text{curr}} \\ 0 \end{bmatrix}.$$

The solution is

$$\hat{x} = \begin{bmatrix} N \\ c^T \end{bmatrix}^{\dagger} \begin{bmatrix} n^{\mathrm{des}} - Nd^{\mathrm{curr}} \\ 0 \end{bmatrix}.$$

We can expand this as

$$\hat{x} = \begin{bmatrix} N^T & c \end{bmatrix} \begin{bmatrix} NN^T & Nc \\ c^T N^T & c^T c \end{bmatrix}^{-1} \begin{bmatrix} n^{\mathrm{des}} - Nd^{\mathrm{curr}} \\ 0 \end{bmatrix}.$$

16.7 *Minimum cost trading to achieve target sector exposures.* A current portfolio is given by the n-vector h^{curr}, with the entries giving the dollar value invested in the n assets. The total value (or net asset value) of the portfolio is $\mathbf{1}^T h^{\mathrm{curr}}$. We seek a new portfolio, given by the n-vector h, with the same total value as h^{curr}. The difference $h - h^{\mathrm{curr}}$ is called the *trade vector*; it gives the amount of each asset (in dollars) that we buy or sell. The n assets are divided into m industry sectors, such as pharmaceuticals or consumer electronics. We let the m-vector s denote the (dollar value) sector exposures to the m sectors. (See exercise 8.13.) These are given by $s = Sh$, where S is the $m \times n$ sector exposure matrix defined by $S_{ij} = 1$ if asset j is in sector i and $S_{ij} = 0$ if asset j is not in sector i. The new portfolio must have a given sector exposure s^{des}. (The given sector exposures are based on forecasts of whether companies in the different sectors will do well or poorly in the future.)

Among all portfolios that have the same value as our current portfolio and achieve the desired exposures, we wish to minimize the trading cost, given by

$$\sum_{i=1}^{n} \kappa_i (h_i - h_i^{\mathrm{curr}})^2,$$

a weighted sum of squares of the asset trades. The weights κ_i are positive. (These depend on the daily trading volumes of assets, as well as other quantities. In general, it is cheaper to trade assets that have high trading volumes.)

Explain how to find h using constrained least squares. Give the KKT equations that you would solve to find h.

Solution. The trading cost can be expressed as

$$\sum_{i=1}^{n} \kappa_i (h_i - h_i^{\mathrm{curr}})^2 = \|Ah - b\|^2$$

if we define

$$A = \begin{bmatrix} \sqrt{\kappa_1} & 0 & \cdots & 0 \\ 0 & \sqrt{\kappa_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sqrt{\kappa_n} \end{bmatrix}, \qquad b = \begin{bmatrix} \sqrt{\kappa_1} h_1^{\mathrm{curr}} \\ \sqrt{\kappa_2} h_2^{\mathrm{curr}} \\ \vdots \\ \sqrt{\kappa_n} h_n^{\mathrm{curr}} \end{bmatrix}.$$

The conditions on h are that $Sh = s^{\mathrm{des}}$ and $\mathbf{1}^T h = \mathbf{1}^T h^{\mathrm{curr}}$. These conditions can be written as $Ch = d$ with

$$C = \begin{bmatrix} S \\ \mathbf{1}^T \end{bmatrix}, \qquad d = \begin{bmatrix} s^{\mathrm{des}} \\ \mathbf{1}^T h^{\mathrm{curr}} \end{bmatrix}.$$

The KKT conditions are

$$\begin{bmatrix} 2\,\mathbf{diag}(\kappa) & S^T & \mathbf{1} \\ S & 0 & 0 \\ \mathbf{1}^T & 0 & 0 \end{bmatrix} \begin{bmatrix} h \\ z \\ w \end{bmatrix} = \begin{bmatrix} 2\,\mathbf{diag}(\kappa) h^{\mathrm{curr}} \\ s^{\mathrm{des}} \\ \mathbf{1}^T h^{\mathrm{curr}} \end{bmatrix}.$$

16.8 *Minimum energy regulator.* We consider a linear dynamical system with dynamics $x_{t+1} = Ax_t + Bu_t$, where the n-vector x_t is the state at time t and the m-vector u_t is the input at time t. We assume that $x = 0$ represents the desired operating point; the goal is to find an input sequence $u_1, \ldots, u_{T-1}$ that results in $x_T = 0$, given the initial state x_1. Choosing an input sequence that takes the state to the desired operating point at time T is called *regulation*.

Find an explicit formula for the sequence of inputs that yields regulation, and minimizes $\|u_1\|^2 + \cdots + \|u_{T-1}\|^2$, in terms of A, B, T, and x_1. This sequence of inputs is called the *minimum energy regulator*.

Hint. Express x_T in terms of x_1, A, the *controllability matrix*

$$C = \begin{bmatrix} A^{T-2}B & A^{T-3}B & \cdots & AB & B \end{bmatrix},$$

and $(u_1, u_2, \ldots, u_{T-1})$ (which is the input sequence stacked). You may assume that C is wide and has linearly independent rows.

Solution. We first find an expression for x_T in terms of A, B, T, x_1:

$$\begin{aligned}
x_2 &= Ax_1 + Bu_1 \\
x_3 &= Ax_2 + Bu_2 \\
&= A^2x_1 + ABu_1 + Bu_2 \\
x_4 &= Ax_3 + Bu_3 \\
&= A^3x_1 + A^2Bu_1 + ABu_2 + Bu_3 \\
&\vdots \\
x_T &= Ax_{T-1} + Bu_{T-1} \\
&= A^{T-1}x_1 + A^{T-2}Bu_1 + \cdots + ABu_{T-2} + Bu_{T-1}.
\end{aligned}$$

This can be expressed as

$$x_T = A^{T-1}x_1 + C\tilde{u}$$

where C is the controllability matrix and $\tilde{u} = (u_1, u_2, \ldots, u_{T-1})$. The condition that $x_T = 0$ is a linear equation

$$C\tilde{u} = -A^{T-1}x_1.$$

The minimum energy regulator minimizes the energy

$$\|u_1\|^2 + \cdots + \|u_{T-1}\|^2 = \|\tilde{u}\|^2.$$

We see that computing the minimum energy regulator is a least norm problem

$$\begin{array}{ll} \text{minimize} & \|\tilde{u}\|^2 \\ \text{subject to} & C\tilde{u} = -A^{T-1}x_1. \end{array}$$

The solution $\tilde{u}$ is the optimal input sequence:

$$(u_1, \ldots, u_{T-1}) = -C^\dagger A^{T-1}x_1.$$

16.9 *Smoothest force sequence to move a mass.* We consider the same setup as the example given on page 343, where the 10-vector f represents a sequence of forces applied to a unit mass over 10 1-second intervals. As in the example, we wish to find a force sequence f that achieves zero final velocity and final position one. In the example on page 343, we choose the smallest f, as measured by its norm (squared). Here, though, we want the *smoothest* force sequence, *i.e.*, the one that minimizes

$$f_1^2 + (f_2 - f_1)^2 + \cdots + (f_{10} - f_9)^2 + f_{10}^2.$$

(This is the sum of the squares of the differences, assuming that $f_0 = 0$ and $f_{11} = 0$.) Explain how to find this force sequence. Plot it, and give a brief comparison with the force sequence found in the example on page 343.

Solution. The constraints on f are the same as on page 343: $Cf = d$, where

$$\begin{bmatrix} 1 & 1 & \cdots & 1 & 1 \\ 19/2 & 17/2 & \cdots & 3/2 & 1/2 \end{bmatrix}, \qquad d = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

The objective is to minimize

$$f_1^2 + (f_2 - f_1)^2 + \cdots + (f_{10} - f_9)^2 + f_{10}^2 = \|Af\|^2$$

where

$$A = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ -1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -1 & 1 \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{bmatrix}.$$

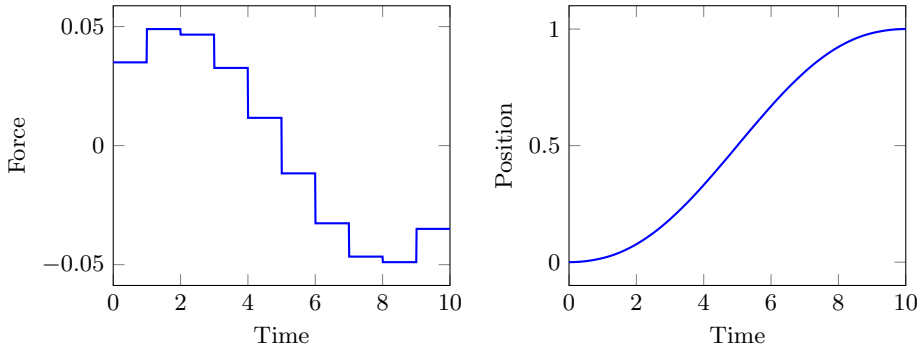
The optimal force sequence can be computed by solving the constrained least squares problem

$$\begin{array}{ll} \text{minimizes} & \|Af\|^2 \\ \text{subject to} & Cf = d. \end{array}$$

The solution is

$f = (0.0350, 0.0490, 0.0466, 0.0326, 0.0117, -0.0117, -0.0326, -0.0466, -0.0490, -0.0350)$.

The force sequence and position are shown in the figures below.



The main difference with the smallest force solution (figure 16.4) are the smaller values at the beginning and the end of the sequence. This makes sense because we have assumed that $f_0 = f_{11} = 0$, so the biggest contribution to the objective are the variations at the beginning and end of the sequence.

16.10 *Smallest force sequence to move a mass to a given position.* We consider the same setup as the example given on page 343, where the 10-vector f represents a sequence of forces applied to a unit mass over 10 1-second intervals. In that example the goal is to find the smallest force sequence (measured by $\|f\|^2$) that achieves zero final velocity and final position one. Here we ask, what is the smallest force sequence that achieves final position one? (We impose no condition on the final velocity.) Explain how to find this force sequence. Compare it to the force sequence found in the example, and give a brief intuitive explanation of the difference. *Remark.* Problems in which the final position of

an object is specified, but the final velocity doesn't matter, generally arise in applications that are not socially positive, for example control of missiles.

Solution. This is a least norm problem: minimize $\|f\|^2$ subject to $a^T f = 1$, where $a = (19/2, 17/2, \ldots, 1/2)$ is the vector for which $p^{\text{fin}} = a^T f$. The solution is

$$\hat{f} = (a^T)^\dagger 1 = a^\dagger = \frac{1}{\|a\|^2} a.$$

Here is the force sequence:

$$\hat{f} = \begin{bmatrix} 0.0285714 \\ 0.0255639 \\ 0.0225564 \\ 0.0195489 \\ 0.0165414 \\ 0.0135338 \\ 0.0105263 \\ 0.0075188 \\ 0.0045112 \\ 0.0015037 \end{bmatrix}.$$

We can see a few effects when comparing this force sequence to the optimal force sequence when there is also a constraint on the final velocity. First, the force is always positive. This makes sense: Negative force is needed only if you insist that the mass arrives at the required position with zero velocity. Second, the force is smaller. That's because we don't use braking (*i.e.*, negative forces). Third, the force is larger earlier on. This makes sense: Forces applied earlier contribute more to the final position, so it makes sense to apply higher forces early on.

16.11 *Least distance problem.* A variation on the least norm problem (16.2) is the least distance problem,

$$\begin{array}{ll} \text{minimize} & \|x - a\|^2 \\ \text{subject to} & Cx = d, \end{array}$$

where the n-vector x is to be determined, the n-vector a is given, the $p \times n$ matrix C is given, and the p-vector d is given. Show that the solution of this problem is

$$\hat{x} = a - C^\dagger(Ca - d),$$

assuming the rows of C are linearly independent. *Hint.* You can argue directly from the KKT equations for the least distance problem, or solve for the variable $y = x - a$ instead of x.

Solution. Following the first option in the hint, we consider the optimality conditions

$$\begin{bmatrix} 2I & C^T \\ C & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} = \begin{bmatrix} 2a \\ d \end{bmatrix}.$$

This gives two equations,

$$2x + C^T z = 2a, \qquad Cx = d.$$

From the first equation, $x = a - (1/2)C^T z$. Substituting this in the second equation gives

$$\frac{1}{2} CC^T z = Ca - d.$$

If C has linearly independent rows, the Gram matrix CC^T is nonsingular, and we get

$$z = 2(CC^T)^{-1}(Ca - d).$$

Substituting this in the expression for x gives the solution

$$\begin{aligned} x &= a - C^T(CC^T)^{-1}(Ca - d) \\ &= a - C^\dagger(Ca - d). \end{aligned}$$

For completeness, we also work out the second approach mentioned in the hint. If we make a change of variable $y = x - a$ the problem becomes

$$\begin{array}{ll} \text{minimize} & \|y\|^2 \\ \text{subject to} & Cy = d - Ca. \end{array}$$

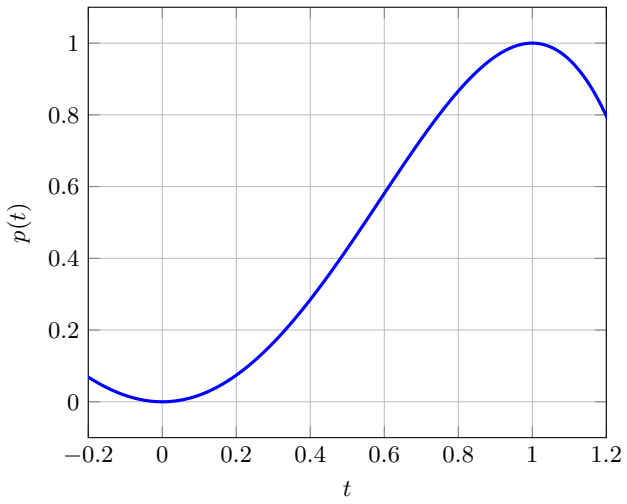
This is a least norm problem with solution $\hat{y} = C^\dagger(d - Ca)$. The optimal solution of the original problem is

$$\hat{x} = a + \hat{y} = a + C^\dagger(d - Ca).$$

16.12 *Least norm polynomial interpolation.* (Continuation of exercise 8.7.) Find the polynomial of degree 4 that satisfies the interpolation conditions given in exercise 8.7, and minimizes the sum of the squares of its coefficients. Plot it, to verify that if satisfies the interpolation conditions.

Solution. The coefficients of the optimal polynomial $p(t) = c_1 + c_2t + c_3t^2 + c_4t^3 + c_5t^4$ are

$$c_1 = 0, \qquad c_2 = 0, \qquad c_3 = 1.8333, \qquad c_4 = 0.3333, \qquad c_5 = -1.1667.$$



16.13 *Steganography via least norm.* In steganography, a secret message is embedded in an image in such a way that the image looks the same, but an accomplice can decode the message. In this exercise we explore a simple approach to steganography that relies on constrained least squares. The secret message is given by a k-vector s with entries that are all either $+1$ or -1 (*i.e.*, it is a Boolean vector). The original image is given by the n-vector x, where n is usually much larger than k. We send (or publish or transmit) the modified message $x + z$, where z is an n-vector of modifications. We would like z to be small, so that the original image x and the modified one $x+z$ look (almost) the same. Our accomplice decodes the message s by multiplying the modified image by a $k \times n$ matrix D, which yields the k-vector $y = D(x + z)$. The message is then decoded as $\hat{s} = \mathbf{sign}(y)$. (We write $\hat{s}$ to show that it is an estimate, and might not be the same as the original.) The matrix D must have linearly independent rows, but otherwise is arbitrary.

(a) *Encoding via least norm.* Let α be a positive constant. We choose z to minimize $\|z\|^2$ subject to $D(x+z) = \alpha s$. (This guarantees that the decoded message is correct, *i.e.*, $\hat{s} = s$.) Give a formula for z in terms of $D^\dagger$, α, and x.

(b) *Complexity.* What is the complexity of encoding a secret message in an image? (You can assume that $D^\dagger$ is already computed and saved.) What is the complexity of decoding the secret message? About how long would each of these take with a computer capable of carrying out 1 Gflop/s, for $k = 128$ and $n = 512^2 = 262144$ (a 512×512 image)?

(c) *Try it out.* Choose an image x, with entries between 0 (black) and 1 (white), and a secret message s with k small compared to n, for example, $k = 128$ for a 512×512 image. (This corresponds to 16 bytes, which can encode 16 characters, *i.e.*, letters, numbers, or punctuation marks.) Choose the entries of D randomly, and compute $D^\dagger$. The modified image $x+z$ may have entries outside the range $[0, 1]$. We replace any negative values in the modified image with zero, and any values greater than one with one. Adjust α until the original and modified images look the same, but the secret message is still decoded correctly. (If α is too small, the clipping of the modified image values, or the round-off errors that occur in the computations, can lead to decoding error, *i.e.*, $\hat{s} \neq s$. If α is too large, the modification will be visually apparent.) Once you've chosen α, send several different secret messages embedded in several different original images.

Solution.

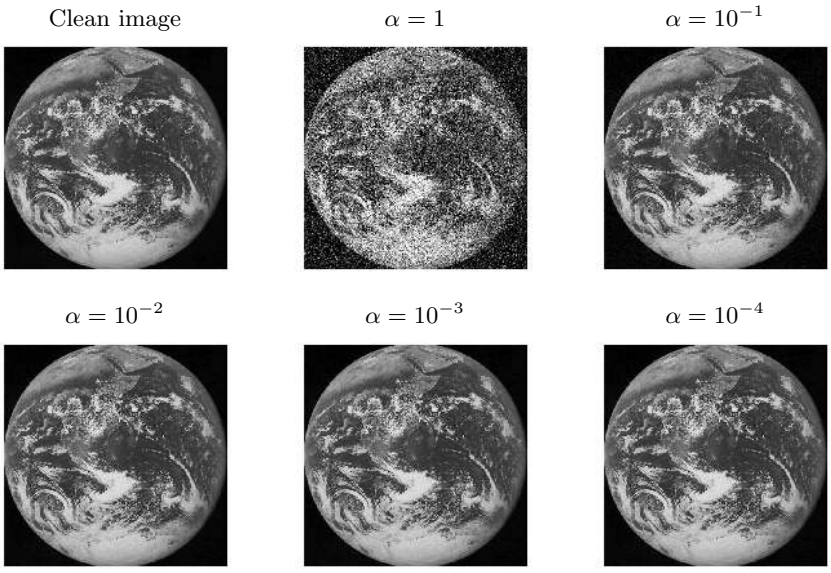
(a) We need to minimize $\|z\|^2$ subject to the equality constraint $D(x+z) = \alpha s$, which we can write as $Dz = \alpha s - Dx$. This is a least norm problem, with solution (assuming that D has linearly independent rows)

$$z = D^\dagger(\alpha s - Dx).$$

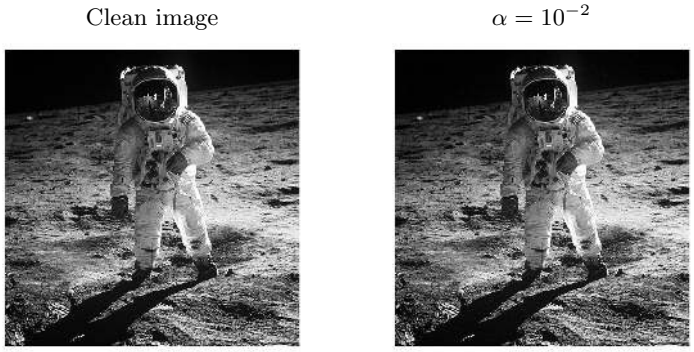
(b) To encode, we compute Dx, which costs $2kn$ flops, subtract this k-vector from αs, which costs k flops, and then multiply by $D^\dagger$, which costs $2kn$ flops. We can ignore the subtraction, so we get around $4kn$ flops. To decode, we multiply by D, which costs $2kn$ flops. (We don't count taking the sign of the result.)

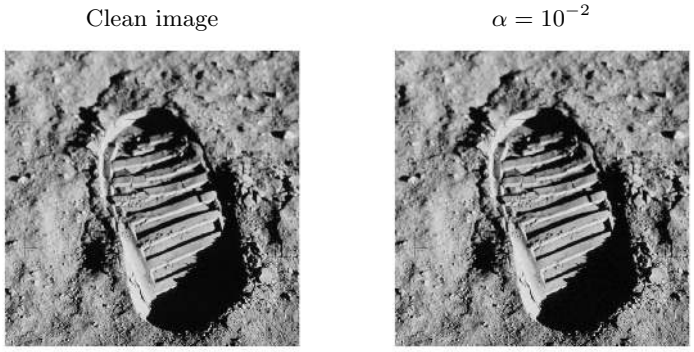
For $k = 128$ and $n = 512^2$ we get $4kn = 3.4 \times 10^7$ flops, which would take around 0.034 seconds. Decoding would take about half that, *i.e.*, around 0.017 seconds.

(c) The example uses a 512×512 image and a message of length $k = 128$. The matrix D is generated randomly and normalized to have norm $\|D\| = 1$. The figure also shows the image with the message encoded and five values of α. For the first three values of α the message was decoded correctly. For $\alpha = 10^{-3}$, the number of incorrectly decoded entries was 3; for $\alpha = 10^{-4}$, 43.



The next figures shows the results for two more images of the same size. A message of length 128 in embedded in the two images on the right, using $\alpha = 10^{-2}$. In both cases the message was decoded correctly.





16.14 *Invertibility of matrix in sparse constrained least squares formulation.* Show that the $(m+n+p)\times(m+n+p)$ coefficient matrix appearing in equation (16.11) is invertible if and only if the KKT matrix is invertible, *i.e.*, the conditions (16.5) hold.

Solution. Suppose the columns of A are linearly independent. Assume that x, y satisfy

$$\begin{bmatrix} 0 & A^T \\ A & I \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Then $A^Ty = 0$ and $Ax + y = 0$. Substituting $y = -Ax$ in the first equation gives $A^TAx = 0$. Therefore $x^TA^TAx = \|Ax\|^2 = 0$, *i.e.*, $Ax = 0$. Since the columns of A are linearly independent, this is only possible if $x = 0$. If $x = 0$, then also $y = -Ax = 0$. We have shown that

$$\begin{bmatrix} 0 & A^T \\ A & I \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

only holds if $x = 0$ and $y = 0$. This means the matrix is invertible.

Next, suppose the columns of A are linearly dependent. Then there exists a nonzero x with $Ax = 0$. We have

$$\begin{bmatrix} 0 & A^T \\ A & I \end{bmatrix}\begin{bmatrix} x \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

and since $(x, 0) \neq 0$ this shows the matrix is not invertible.

16.15 *Approximating each column of a matrix as a linear combination of the others.* Suppose A is an $m \times n$ matrix with linearly independent columns $a_1, \ldots, a_n$. For each i we consider the problem of finding the linear combination of $a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n$ that is closest to a_i. These are n standard least squares problems, which can be solved using the methods of chapter 12. In this exercise we explore a simple formula that allows us to solve these n least squares problem all at once. Let $G = A^TA$ denote the Gram matrix, and $H = G^{-1}$ its inverse, with columns $h_1, \ldots, h_n$.

(a) Explain why minimizing $\|Ax^{(i)}\|^2$ subject to $x_i^{(i)} = -1$ solves the problem of finding the linear combination of $a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n$ that is closest to a_i. These are n constrained least squares problems.

(b) Solve the KKT equations for these constrained least squares problems,

$$\begin{bmatrix} 2A^TA & e_i \\ e_i^T & 0 \end{bmatrix}\begin{bmatrix} x^{(i)} \\ z_i \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \end{bmatrix},$$

to conclude that $x^{(i)} = -(1/H_{ii})h_i$. In words: $x^{(i)}$ is the ith column of $(A^TA)^{-1}$, scaled so its ith entry is -1.

(c) Each of the n original least squares problems has $n-1$ variables, so the complexity is $n(2m(n-1)^2)$ flops, which we can approximate as $2mn^3$ flops. Compare this to the complexity of a method based on the result of part (b): First find the QR factorization of A; then compute H.

(d) Let d_i denote the distance between a_i and the linear combination of the other columns that is closest to it. Show that $d_i = 1/\sqrt{H_{ii}}$.

Remark. When the matrix A is a data matrix, with A_{ij} the value of the jth feature on the ith example, the problem addressed here is the problem of predicting each of the features from the others. The numbers d_i tells us how well each feature can be predicted from the others.

Solution.

(a) In the ith least squares problem we minimize

$$\|\sum_{j\neq i} a_j x_j^{(i)} - a_i\|^2.$$

The variables are the $n-1$ coefficients $x_j^{(i)}$ for $j \neq i$. This is equivalent to minimizing

$$\|\sum_{j=1}^{n} a_j x_j^{(i)}\|^2$$

with the constraint that $x^{(i)} = -1$.

(b) From the first equation, we see that

$$x^{(i)} = -\frac{z_i}{2}(A^T A)^{-1} e_i = -\frac{z_i}{2} h_i.$$

The value of z_i follows from the equation $x_i^{(i)} = -1$:

$$x_i^{(i)} = -\frac{z_i}{2} H_{ii} = -1$$

if $z_i = 2/H_{ii}$. Substituting this in the expression for $x^{(i)}$ gives $x^{(i)} = -(1/H_{ii})h_i$.

(c) Computing the QR factorization of A take $2mn^2$ flops. The matrix H is $H = R^{-1}(R^{-1})^T$ costs order n^3 flops, order n^3 for finding R^{-1} and order n^3 for the product.

(d) The squared distance is equal to

$$\|Ax^{(i)}\|^2 = \frac{1}{H_{ii}^2}\|AHe_i\|^2 = \frac{1}{H_{ii}^2}(e_i^T H A^T A H e_i) = \frac{1}{H_{ii}}.$$

Chapter 17

Constrained least squares applications

Exercises

17.1 *A variation on the portfolio optimization formulation.* Consider the following variation on the linearly constrained least squares problem (17.2):

$$\begin{array}{ll} \text{minimize} & \|Rw\|^2 \\ \text{subject to} & \begin{bmatrix} \mathbf{1}^T \\ \mu^T \end{bmatrix} w = \begin{bmatrix} 1 \\ \rho \end{bmatrix}, \end{array} \tag{17.1}$$

with variable w. (The difference is that here we drop the term $\rho\mathbf{1}$ that appears inside the norm square objective in (17.2).) Show that this problem is equivalent to (17.2). This means w is a solution of (17.1) if and only if it is a solution of (17.2).

Hint. You can argue directly by expanding the objective in (17.2) or via the KKT systems of the two problems.

Solution. The objective in (17.2) is

$$\|Rw - \rho\mathbf{1}\|^2 = \|Rw\|^2 - 2\rho w^T R^T \mathbf{1} + \rho T = \|Rw\|^2 - 2\rho T w^T \mu + \rho T$$

The second step follows from $\mu = R^T\mathbf{1}/T$.

If w satisfies $\mu^T w = 1$, as it does in the two variations of the problem, we can simplify this further:

$$\|Rw - \rho\mathbf{1}\|^2 = \|Rw\|^2 - \rho T.$$

This shows the two objectives differ only by constant, so the optimal w is the same.

17.2 *A more conventional formulation of the portfolio optimization problem.* In this problem we derive an equivalent formulation of the portfolio optimization problem (17.2) that appears more frequently in the literature than our version. (Equivalent means that the two problems always have the same solution.) This formulation is based on the *return covariance matrix*, which we define below. (See also exercise 10.16.)

The means of the columns of the asset return matrix R are the entries of the vector μ. The *de-meaned returns matrix* is given by $\tilde{R} = R - \mathbf{1}\mu^T$. (The columns of the matrix $\tilde{R} = R - \mathbf{1}\mu^T$ are the de-meaned return time series for the assets.) The return covariance matrix, traditionally denoted Σ, is its Gram matrix $\Sigma = (1/T)\tilde{R}^T\tilde{R}$.

(a) Show that $\sigma_i = \sqrt{\Sigma_{ii}}$ is the standard deviation (risk) of asset i return. (The symbol σ_i is a traditional one for the standard deviation of asset i.)

(b) Show that the correlation coefficient between asset i and asset j returns is given by $\rho_{ij} = \Sigma_{ij}/(\sigma_i\sigma_j)$. (Assuming neither asset has constant return; if either one does, they are uncorrelated.)

(c) *Portfolio optimization using the return covariance matrix.* Show that the following problem is equivalent to our portfolio optimization problem (17.2):

$$\begin{array}{ll} \text{minimize} & w^T\Sigma w \\ \text{subject to} & \begin{bmatrix} \mathbf{1}^T \\ \mu^T \end{bmatrix} w = \begin{bmatrix} 1 \\ \rho \end{bmatrix}, \end{array} \tag{17.2}$$

with variable w. This is the form of the portfolio optimization problem that you will find in the literature. *Hint.* Show that the objective is the same as $\|\tilde{R}w\|^2$, and that this is the same as $\|Rw - \rho\mathbf{1}\|^2$ for any feasible w.

Solution.

(a) The ith diagonal entry of Σ is

$$\Sigma_{ii} = \frac{1}{T} e_i^T \tilde{R}\tilde{R} e_i = \mathbf{rms}(\tilde{R}e_i)^2 = \mathbf{std}(Re_i)^2.$$

This shows that $\sigma_i = \mathbf{std}(Re_i)$, the standard deviation of column i of R, *i.e.*, the risk of asset i.

(b) The correlation coefficient between columns i and j of R is defined as

$$\rho_{ij} = \frac{1}{T}\frac{(\tilde{R}e_i)^T(\tilde{R}e_j)}{\mathbf{std}(Re_i)\,\mathbf{std}(\tilde{R}e_j)} = \frac{1}{T}\frac{e_i^T\tilde{R}^T\tilde{R}e_j}{\mathbf{std}(Re_i)\,\mathbf{std}(\tilde{R}e_j)} = \frac{\Sigma_{ij}}{\sigma_i\sigma_j}.$$

(c) The objective in (17.2) is

$$\|Rw - \rho\mathbf{1}\|^2 = \|(\tilde{R} + \mathbf{1}\mu^T)w - \rho\mathbf{1}\|^2 = \|\tilde{R}w - (\rho - \mu^T w)\mathbf{1}\|^2.$$

Problems (17.2) and (17.2) both include a constraint that $\mu^T w = \rho$. If w satisfies this condition, the objective further simplifies to

$$\|\tilde{R}w\|^2 = w^T\tilde{R}^T\tilde{R}w = (w^T\Sigma w)/T.$$

Up to a constant factor T the two problems have the same objective.

17.3 *A simple portfolio optimization problem.*

(a) Find an analytical solution for the portfolio optimization problem with $n = 2$ assets. You can assume that $\mu_1 \neq \mu_2$, *i.e.*, the two assets have different mean returns. *Hint.* The optimal weights depend only on μ and ρ, and not (directly) on the return matrix R.

(b) Find the conditions under which the optimal portfolio takes long positions in both assets, a short position in one and a long position in the other, or a short position in both assets. You can assume that $\mu_1 < \mu_2$, *i.e.*, asset 2 has the higher return. *Hint.* Your answer should depend on whether $\rho < \mu_1$, $\mu_1 < \rho < \mu_2$, or $\mu_2 < \rho$, *i.e.*, how the required return compares to the two asset returns.

Solution. The two constraints determine w_1 and w_2 uniquely:

$$w_1 + w_2 = 1, \qquad \mu_1 w_1 + \mu_2 w_2 = \rho.$$

The solution is

$$w_1 = \frac{\mu_2 - \rho}{\mu_2 - \mu_1}, \qquad w_2 = \frac{\rho - \mu_1}{\mu_2 - \mu_1}.$$

Let's assume that $\mu_1 < \mu_2$, *i.e.*, asset 2 is the one with higher return. (If this isn't the case, just switch around what we call asset 1 and asset 2.) We will refer to μ_2 as the larger return, and μ_1 as the smaller return. The denominators of the expressions for w_1 and w_2 above are positive, so the signs of w_1 and w_2 are the signs of the numerators.

We see that we take a long position in the lower return asset (*i.e.*, $w_1 > 0$) when $\rho < \mu_2$, *i.e.*, the required return is less than the larger return. We take a long position in asset 2, the higher return asset, when $\rho > \mu_1$, *i.e.*, the required return is larger than the smaller asset return.

We can summarize this as follows:

- $\rho < \mu_1 < \mu_2$: Long position in asset 1, short position in asset 2.
- $\mu_1 < \rho < \mu_2$: Long position in asset 1, long position in asset 2.
- $\mu_1 < \mu_2 < \rho$: Short positions in asset 1, long position in asset 2.

We never take a short position in both assets. Actually, that's clear from $w_1 + w_2 = 1$.

17.4 *Index tracking.* Index tracking is a variation on the portfolio optimization problem described in §17.1. As in that problem we choose a portfolio allocation weight vector w that satisfies $\mathbf{1}^T w = 1$. This weight vector gives a portfolio return time series Rw, which is a T-vector. In index tracking, the goal is for this return time series to track (or follow) as closely as possible a given target return time series r^{tar}. We choose w to minimize the RMS deviation between the target return time series r^{tar} and the portfolio return time series r. (Typically the target return is the return of an index, like the Dow Jones

Industrial Average or the Russell 3000.) Formulate the index tracking problem as a linearly constrained least squares problem, analogous to (17.2). Give an explicit solution, analogous to (17.3).

Solution. The index tracking problem can be written as a constrained least squares problem

$$\begin{array}{ll} \text{minimize} & \|Rw - r^{\text{tar}}\|^2 \\ \text{subject to} & \mathbf{1}^T w = 1. \end{array}$$

The analog of 17.3) is

$$\begin{bmatrix} w \\ z \end{bmatrix} = \begin{bmatrix} 2R^TR & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix}^{-1} \begin{bmatrix} 2R^Tr^{\text{tar}} \\ 1 \end{bmatrix}.$$

17.5 *Portfolio optimization with market neutral constraint.* In the portfolio optimization problem (17.2) the portfolio return time series is the T-vector Rw. Let r^{mkt} denote the T-vector that gives the return of the whole market over the time periods $t = 1, \ldots, T$. (This is the return associated with the total value of the market, *i.e.*, the sum over the assets of asset share price times number of outstanding shares.) A portfolio is said to be *market neutral* if Rw and r^{mkt} are uncorrelated.

Explain how to formulate the portfolio optimization problem, with the additional constraint of market neutrality, as a constrained least squares problem. Give an explicit solution, analogous to (17.3).

Solution. The portfolio optimization problem (17.2) includes a constraint that $\mathbf{avg}(Rw) = \rho$. Therefore the demeaned portfolio return vector is

$$Rw - \mathbf{avg}(Rw)\mathbf{1} = Rw - \rho\mathbf{1}.$$

The demeaned market return vector is

$$\tilde{r}^{\text{mkt}} = r^{\text{mkt}} - \mathbf{avg}(r^{\text{mkt}})\mathbf{1}.$$

Market neutrality holds if the two demeaned vectors are orthogonal:

$$\begin{array}{rcl} 0 & = & (r^{\text{mkt}} - \mathbf{avg}(r^{\text{mkt}})\mathbf{1})^T(Rw - \rho\mathbf{1}) \\ & = & (r^{\text{mkt}} - \mathbf{avg}(r^{\text{mkt}})\mathbf{1})^T Rw \\ & = & (r^{\text{mkt}})^T Rw - \mathbf{avg}(r^{\text{mkt}})\mathbf{1}^T Rw \\ & = & (r^{\text{mkt}})^T Rw - T\,\mathbf{avg}(r^{\text{mkt}})\mu^T w \\ & = & (r^{\text{mkt}})^T Rw - T\,\mathbf{avg}(r^{\text{mkt}})\rho. \end{array}$$

On line 4 we use the definition $\mu = (1/T)R^T\mathbf{1}$. The last line follows from the constraint $\mu^T w = \rho$.

Adding the market neutrality constraint to (17.2) gives

$$\begin{array}{ll} \text{minimize} & \|Rw - \rho\mathbf{1}\|^2 \\ \text{subject to} & \begin{bmatrix} \mathbf{1}^T \\ \mu^T \\ (r^{\text{mkt}})^T R \end{bmatrix} w = \begin{bmatrix} 1 \\ \rho \\ \rho T\,\mathbf{avg}(r^{\text{mkt}}) \end{bmatrix}. \end{array}$$

The solution can be expressed as

$$\begin{bmatrix} w \\ z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} 2R^TR & \mathbf{1} & \mu & R^Tr^{\text{mkt}} \\ \mathbf{1}^T & 0 & 0 & 0 \\ \mu^T & 0 & 0 & 0 \\ (R^Tr^{\text{mkt}})^T & 0 & 0 & 0 \end{bmatrix}^{-1} \begin{bmatrix} 2\rho T\mu \\ 1 \\ \rho \\ \rho T\,\mathbf{avg}(r^{\text{mkt}}) \end{bmatrix}.$$

17.6 *State feedback control of the longitudinal motions of a Boeing 747 aircraft.* In this exercise we consider the control of the longitudinal motions of a Boeing 747 aircraft in steady level flight, at an altitude of 40000 ft, and speed 774 ft/s, which is 528 MPH or 460 knots, around Mach 0.8 at that altitude. (Longitudinal means that we consider climb rate and speed, but not turning or rolling motions.) For modest deviations from these steady state or *trim* conditions, the dynamics is given by the linear dynamical system $x_{t+1} = Ax_t + Bu_t$, with

$$A = \begin{bmatrix} 0.99 & 0.03 & -0.02 & -0.32 \\ 0.01 & 0.47 & 4.70 & 0.00 \\ 0.02 & -0.06 & 0.40 & 0.00 \\ 0.01 & -0.04 & 0.72 & 0.99 \end{bmatrix}, \qquad B = \begin{bmatrix} 0.01 & 0.99 \\ -3.44 & 1.66 \\ -0.83 & 0.44 \\ -0.47 & 0.25 \end{bmatrix},$$

with time unit one second. The state 4-vector x_t consists of deviations from the trim conditions of the following quantities.

- $(x_t)_1$ is the velocity along the airplane body axis, in ft/s, with forward motion positive.
- $(x_t)_2$ is the velocity perpendicular to the body axis, in ft/s, with positive down.
- $(x_t)_3$ is the angle of the body axis above horizontal, in units of 0.01 radian (0.57°).
- $(x_t)_4$ is the derivative of the angle of the body axis, called the *pitch rate*, in units of 0.01 radian/s (0.57°/s).

The input 2-vector u_t (which we can control) consists of deviations from the trim conditions of the following quantities.

- $(u_t)_1$ is the elevator (control surface) angle, in units of 0.01 radian.
- $(u_t)_2$ is the engine thrust, in units of 10000 lbs.

You do not need to know these details; we mention them only so you know what the entries of x_t and u_t mean.

(a) *Open loop trajectory.* Simulate the motion of the Boeing 747 with initial condition $x_1 = e_4$, in open-loop (*i.e.*, with $u_t = 0$). Plot the state variables over the time interval $t = 1, \ldots, 120$ (two minutes). The oscillation you will see in the open-loop simulation is well known to pilots, and called the *phugoid mode*.

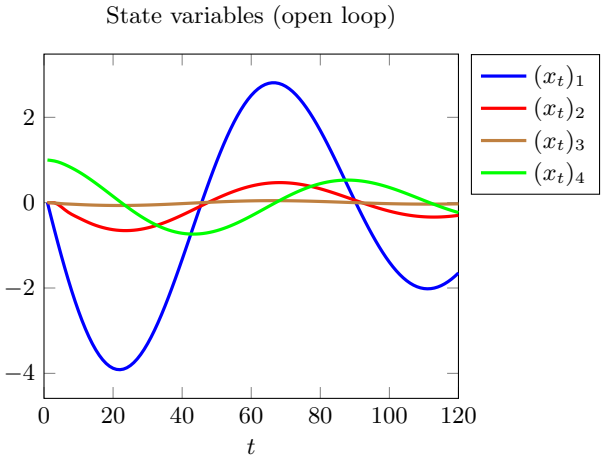
(b) *Linear quadratic control.* Solve the linear quadratic control problem with $C = I$, $\rho = 100$, and $T = 100$, with initial state $x_1 = e_4$, and desired terminal state $x^{\mathrm{des}} = 0$. Plot the state and input variables over $t = 1, \ldots, 120$. (For $t = 100, \ldots, 120$, the state and input variables are zero.)

(c) Find the 2×4 state feedback gain K obtained by solving the linear quadratic control problem with $C = I$, $\rho = 100$, $T = 100$, as described in §17.2.3. Verify that it is almost the same as the one obtained with $T = 50$.

(d) Simulate the motion of the Boeing 747 with initial condition $x_1 = e_4$, under state feedback control (*i.e.*, with $u_t = Kx_t$). Plot the state and input variables over the time interval $t = 1, \ldots, 120$.

Solution.

(a) The figure shows the four states in open loop, starting at $x^{\mathrm{init}} = e_4$.



(b) We define $\tilde{A}$ and $\tilde{C}$ as described in §17.2.3. The matrix $\tilde{A}$ is

$$\tilde{A} = \begin{bmatrix} C & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & C & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & C & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & \sqrt{\rho} I_2 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & \sqrt{\rho} I_2 \end{bmatrix}$$

The matrix $\tilde{A}$ is block-diagonal, with T diagonal blocks C and $T-1$ diagonal blocks $\sqrt{\rho} I_2$. Therefore $\tilde{A}$ is square with $(4T + 2(T-1)) = 6T - 2$ rows and columns. The matrix $\tilde{C}$ and the right-hand side $\tilde{d}$ are defined as

$$\tilde{C} = \begin{bmatrix} A & -I_4 & 0 & \cdots & 0 & 0 & B & 0 & \cdots & 0 \\ 0 & A & -I_4 & \cdots & 0 & 0 & 0 & B & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & A & -I_4 & 0 & 0 & \cdots & B \\ I_4 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 & I_4 & 0 & 0 & \cdots & 0 \end{bmatrix}, \qquad \tilde{d} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ x^{\text{init}} \\ 0 \end{bmatrix}.$$

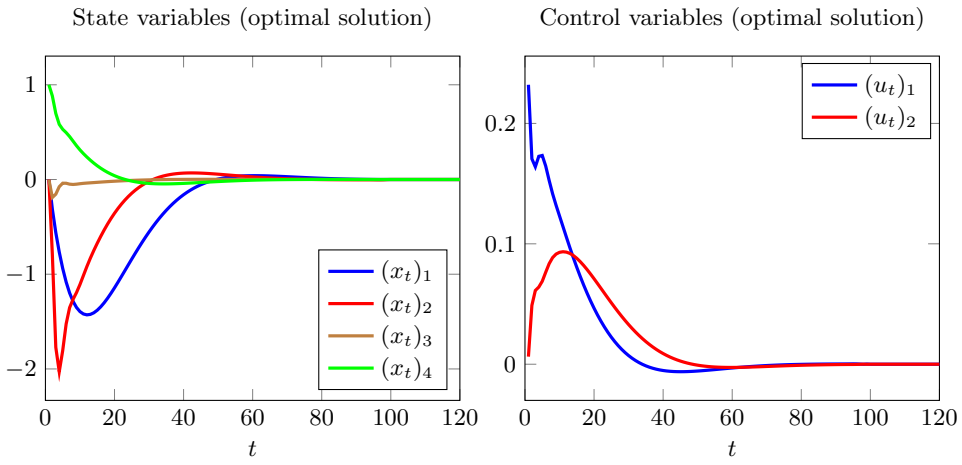
There are $T-1$ matrices A and $T-1$ matrices B in the definition of $\tilde{C}$. Therefore $\tilde{C}$ has $4(T-1)+8 = 4T+4$ rows and $4T + 2(T-1) = 6T - 2$ columns.

The constrained least squares problem is

$$\begin{array}{ll} \text{minimize} & \|\tilde{A}\tilde{x}\|^2 \\ \text{subject to} & \tilde{C}\tilde{x} = \tilde{d}. \end{array}$$

with a variable $\tilde{x} = (x_1, \ldots, x_T, u_1, \ldots, u_{T-1})$ of size $4T + 2(T-1) = 6T - 2$.

The next two figures show the optimal solution of the linear quadratic control problem for initial state e_4 and $T = 100$. We simulate for 120 intervals, assuming the control variable remains zero after interval $T - 1 = 99$.



(c) We note that $\tilde{d}$ depends on linearly on x^{init}. If we define

$$\tilde{D} = \begin{bmatrix} 0_{4\times 4} \\ 0_{4\times 4} \\ \vdots \\ 0_{4\times 4} \\ I_4 \\ 0_{4\times 4} \end{bmatrix}$$

we can write $\tilde{d} = \tilde{D}x^{\text{init}}$. This allows us to express the solution of the linear quadratic control problem as a linear function of x^{init}:

$$\begin{bmatrix} \tilde{x} \\ \tilde{z} \end{bmatrix} = \begin{bmatrix} \tilde{X} \\ \tilde{Z} \end{bmatrix} x^{\text{init}}$$

where

$$\begin{bmatrix} \tilde{X} \\ \tilde{Z} \end{bmatrix} = \begin{bmatrix} 2\tilde{A}^T\tilde{A} & \tilde{C}^T \\ \tilde{C} & 0 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ \tilde{D} \end{bmatrix}.$$

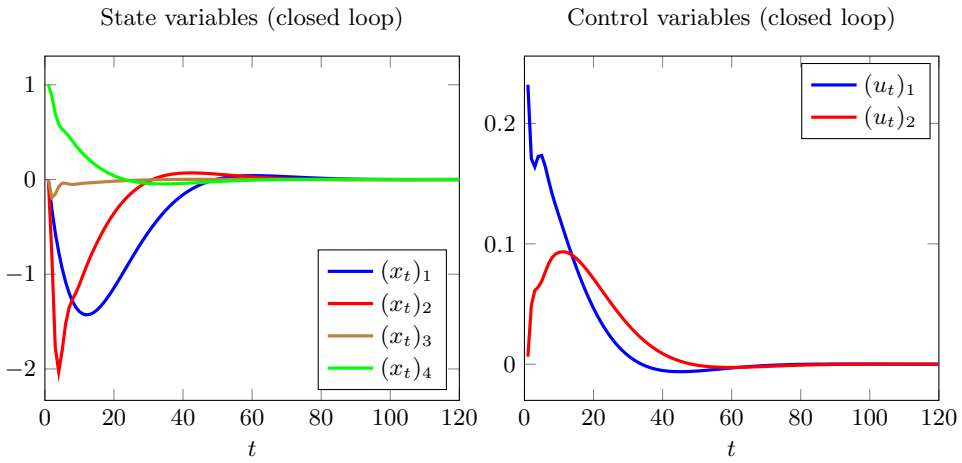
In particular, $u_1 = \tilde{x}_{4T+1:4T+2} = Kx^{\text{init}}$ where $K = \tilde{X}_{4T+1:4T+2,:}$. For $T = 50$, we obtain

$$K = \begin{bmatrix} -0.03179 & -0.01805 & 0.30718 & 0.23247 \\ -0.05916 & -0.00302 & -0.11855 & 0.00611 \end{bmatrix}$$

and for $T = 100$,

$$K = \begin{bmatrix} -0.03177 & -0.01801 & 0.30710 & 0.23209 \\ -0.05904 & -0.00304 & -0.11852 & 0.00630 \end{bmatrix}.$$

(d) The last two figures show the state and control variables under state feedback control, simulated for 120 intervals. As can be seen by comparing it with the previous plots, the state feedback control is very close to optimal.



17.7 *Bio-mass estimation.* A bio-reactor is used to grow three different bacteria. We let x_t be the 3-vector of the bio-masses of the three bacteria, at time period (say, hour) t, for $t = 1, \ldots, T$. We believe that they each grow, independently, with growth rates given by the 3-vector r (which has positive entries). This means that $(x_{t+1})_i \approx (1 + r_i)(x_t)_i$, for $i = 1, 2, 3$. (These equations are approximate; the real rate is not constant.) At every time sample we measure the total bio-mass in the reactor, *i.e.*, we have measurements $y_t \approx \mathbf{1}^T x_t$, for $t = 1, \ldots, T$. (The measurements are not exactly equal to the total mass; there are small measurement errors.) We do not know the bio-masses $x_1, \ldots, x_T$, but wish to estimate them based on the measurements $y_1, \ldots, y_T$.

Set this up as a linear quadratic state estimation problem as in §17.3. Identify the matrices A_t, B_t, and C_t. Explain what effect the parameter λ has on the estimated bio-mass trajectory $\hat{x}_1, \ldots, \hat{x}_T$.

Solution. We take

$$A = \mathbf{diag}(\mathbf{1} + r), \quad B = I, \quad C = \mathbf{1}^T.$$

(These do not depend on t.) The process error w_t is the difference between the actual next bio-mass vector and the one predicted by our simple growth model, *i.e.*, $w_t = x_{t+1} - Ax_t$. The measurement error v_t is the difference between our measurement y_t and the actual total bio-mass $\mathbf{1}^T x_t$.

Linear state estimation will choose state estimates $\hat{x}_1, \ldots, \hat{x}_T$ to minimize a weighted sum of squares of process and noise errors. When λ is large, we trust our measurements, and ascribe errors mostly to the dynamics equation; when λ is small, we don't trust our measurements, and ascribe errors mostly to the measurements.

Chapter 18

Nonlinear least squares

Exercises

18.1 *Lambert W-function.* The *Lambert W-function*, denoted $W : [0, \infty) \to \mathbf{R}$, is defined as $W(u) = x$, where x is the unique number $x \geq 0$ for which $xe^x = u$. (The notation just means that we restrict the argument x to be nonnegative.) The Lambert function arises in a variety of applications, and is named after the mathematician Johann Heinrich Lambert. There is no analytical formula for $W(u)$; it must be computed numerically. In this exercise you will develop a solver to compute $W(u)$, given a nonnegative number u, using the Levenberg–Marquardt algorithm, by minimizing $f(x)^2$ over x, where $f(x) = xe^x - u$.

(a) Give the Levenberg–Marquardt update (18.13) for f.

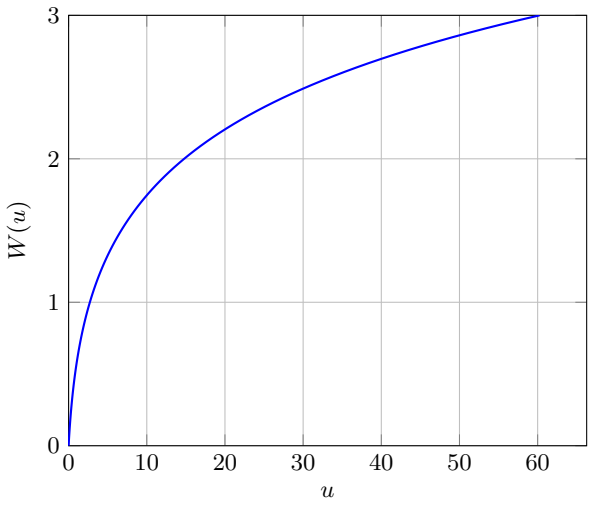
(b) Implement the Levenberg–Marquardt algorithm for minimizing $f(x)^2$. You can start with $x^{(1)} = 1$ and $\lambda^{(1)} = 1$ (but it should work with other initializations). You can stop the algorithm when $|f(x)|$ is small, say, less than 10^{-6}.

Solution.

(a) The derivative of f is $f'(x) = (1+x)e^x$. The Levenberg–Marquardt iteration update is

$$\begin{aligned} x^{(k+1)} &= x^{(k)} - \frac{f'(x^{(k)})}{\lambda^{(k)} + (f'(x^{(k)}))^2} f(x^{(k)}) \\ &= x^{(k)} - \frac{(1 + x^{(k)}) \exp(x^{(k)})}{\lambda^{(k)} + (1 + x^{(k)})^2 \exp(2x^{(k)})} (x^{(k)} \exp(x^{(k)}) - u). \end{aligned}$$

(b) The plot shows the Lambert function.



If we start the Levenberg–Marquardt algorithm at $x^{(1)} = 1$ and $\lambda^{(k)}$, then the values

$$W(5) = 1.3267, \quad W(10) = 1.7455, \quad W(20) = 2.2050$$

are computed in 5, 9, and 11 iterations, respectively.

18.2 *Internal rate of return.* Let the n-vector c denote a cash flow over n time periods, with positive entries meaning cash received and negative entries meaning payments. We assume that its NPV (net present value; see page 22) with interest rate $r \geq 0$ is given by

$$N(r) = \sum_{i=1}^{n} \frac{c_i}{(1 + r)^{i-1}}.$$

The *internal rate of return* (IRR) of the cash flow is defined as the smallest positive value of r for which $N(r) = 0$. The Levenberg–Marquardt algorithm can be used to compute the IRR of a given cash flow sequence, by minimizing $N(r)^2$ over r.

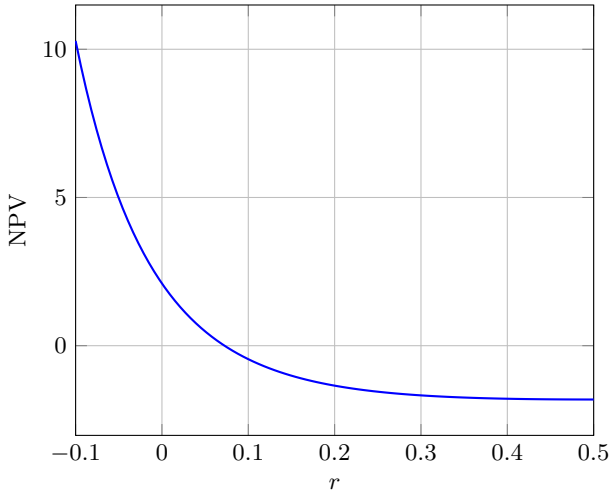
(a) Work out a specific formula for the Levenberg–Marquardt update for r, *i.e.*, (18.13).

(b) Implement the Levenberg–Marquardt algorithm to find the IRR of the cash flow sequence

$$c = (-\mathbf{1}_3,\ 0.3\,\mathbf{1}_5,\ 0.6\,\mathbf{1}_6),$$

where the subscripts give the dimensions. (This corresponds to three periods in which you make investments, which pay off at one rate for 5 periods, and a higher rate for the next 6 periods.) You can initialize with $r^{(0)} = 0$, and stop when $N(r^{(k)})^2$ is small. Plot $N(r^{(k)})^2$ versus k.

Solution. The figure shows the net present value as a function r.



(a) The derivative of $N(r)$ is $N'(r) = \sum_{i=1}^{n}(1-i)c_i/(1+r)^i$. The Levenberg–Marquardt update is

$$r^{(k+1)} = r^{(k)} - \frac{N'(r^{(k)})}{\lambda^{(k)} + (N'(r^{(k)}))^2} N(r^{(k)}).$$

(b) The IRR is 0.0725. The Levenberg–Marquardt method started with $r^{(1)} = 0$ and $\lambda^{(1)} = 1$ converges to a very accurate solution in about 3 or 4 steps. Newton's method, *i.e.*, the simpler update

$$r^{(k+1)} = r^{(k)} - \frac{N(r^{(k)})}{N'(r^{(k)})},$$

also converges very quickly.

18.3 *A common form for the residual.* In many nonlinear least squares problems the residual function $f : \mathbf{R}^n \to \mathbf{R}^m$ has the specific form

$$f_i(x) = \phi_i(a_i^T x - b_i), \quad i = 1, \ldots, m,$$

where a_i is an n-vector, b_i is a scalar, and $\phi_i : \mathbf{R} \to \mathbf{R}$ is a scalar-valued function of a scalar. In other words, $f_i(x)$ is a scalar function of an affine function of x. In this case the objective of the nonlinear least squares problem has the form

$$\|f(x)\|^2 = \sum_{i=1}^{m} \left(\phi_i(a_i^T x - b_i)\right)^2.$$

We define the $m \times n$ matrix A to have rows $a_1^T, \ldots, a_m^T$, and the m-vector b to have entries $b_1, \ldots, b_m$. Note that if the functions ϕ_i are the identity function, *i.e.*, $\phi_i(u) = u$ for all u, then the objective becomes $\|Ax-b\|^2$, and in this case the nonlinear least squares problem reduces to the linear least squares problem. Show that the derivative matrix $Df(x)$ has the form

$$Df(x) = \mathbf{diag}(d)A,$$

where $d_i = \phi_i'(r_i)$ for $i = 1, \ldots, m$, with $r = Ax - b$.

Remark. This means that in each iteration of the Gauss–Newton method we solve a weighted least squares problem (and in the Levenberg–Marquardt algorithm, a regularized weighted least squares problem); see exercise 12.4. The weights change in each iteration.

Solution. The gradient of $f_i(x)$ is

$$\nabla f_i(x) = \phi_i'(a_i^T x - b_i)a_i.$$

The derivative matrix $Df(x)$ has the gradients as its rows:

$$Df(x) = \begin{bmatrix} \phi_1'(a_1^T x - b_1)a_1^T \\ \phi_2'(a_2^T x - b_2)a_2^T \\ \vdots \\ \phi_m'(a_m^T x - b_m)a_m^T \end{bmatrix} = \mathbf{diag}(d)A.$$

18.4 *Fitting an exponential to data.* Use the Levenberg–Marquardt algorithm to fit an exponential function of the form $\hat{f}(x;\theta) = \theta_1 e^{\theta_2 x}$ to the data

$$0,\ 1,\ \ldots,\ 5, \qquad 5.2,\ 4.5,\ 2.7,\ 2.5,\ 2.1,\ 1.9.$$

(The first list gives $x^{(i)}$; the second list gives $y^{(i)}$.) Plot your model $\hat{f}(x;\hat{\theta})$ versus x, along with the data points.

Solution. We minimize the sum of the squares of the prediction residuals

$$\sum_{i=1}^{6} (\theta_1 e^{\theta_2 x_i} - y_i)^2 = \|f(\theta)\|^2$$

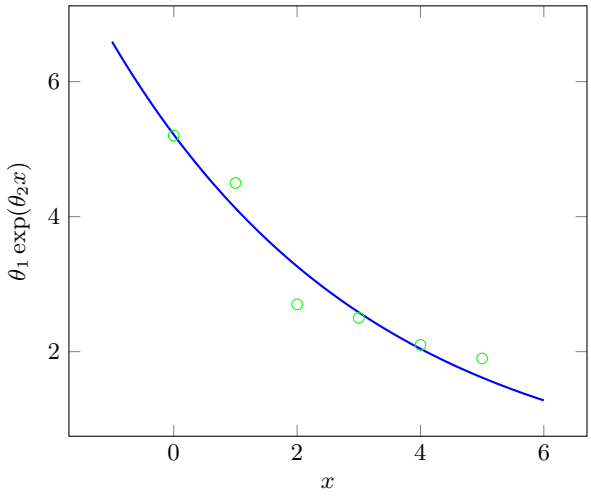
where $f_i(\theta) = \theta_1 \exp(\theta_2 x_i) - y_i$ for $i = 1, \ldots, 6$. The derivative matrix is

$$Df(\theta) = \begin{bmatrix} \exp(\theta_2 x_1) & x_1\theta_1 \exp(\theta_2 x_1) \\ \exp(\theta_2 x_2) & x_2\theta_1 \exp(\theta_2 x_2) \\ \vdots & \vdots \\ \exp(\theta_2 x_6) & x_6\theta_1 \exp(\theta_2 x_6) \end{bmatrix}.$$

We start the Levenberg–Marquardt method at $\theta_1^{(1)} = \theta_2^{(1)} = 0$ and $\lambda^{(1)} = 1$, and stop it when $\|2Df(\theta)^T f(\theta)\| < 10^{-5}$. The algorithm takes 9 steps and returns

$$\theta_1 = 5.2140, \qquad \theta_2 = -0.2343.$$

The solution is shown in the figure.



18.5 *Mechanical equilibrium.* A mass m, at position given by the 2-vector x, is subject to three forces acting on it. The first force F^{grav} is gravity, which has value $F^{\text{grav}} = -mg(0, 1)$, where $g = 9.8$ is the acceleration of gravity. (This force points straight down, with a force that does not depend on the position x.) The mass is attached to two cables, whose other ends are anchored at (2-vector) locations a_1 and a_2. The force F_i applied to the mass by cable i is given by

$$F_i = T_i(a_i - x)/\|a_i - x\|,$$

where T_i is the cable tension. (This means that each cable applies a force on the mass that points from the mass to the cable anchor, with magnitude given by the tension T_i.) The cable tensions are given by

$$T_i = k\frac{\max\{\|a_i - x\| - L_i, 0\}}{L_i},$$

where k is a positive constant, and L_i is the natural or unloaded length of cable i (also positive). In words: The tension is proportional to the fractional stretch of the cable, above its natural length. (The max appearing in this formula means the tension is not a differentiable function of the position, when $\|a_i - x\| = L_i$, but we will simply ignore this.) The mass is in equilibrium at position x if the three forces acting on it sum to zero,

$$F^{\text{grav}} + F_1 + F_2 = 0.$$

We refer to the left-hand side as the residual force. It is a function of mass position x, and we write it as $f(x)$.

Compute an equilibrium position for

$$a_1 = (3, 2), \qquad a_2 = (-1, 1), \qquad L_1 = 3, \qquad L_2 = 2, \qquad m = 1, \qquad k = 100,$$

by applying the Levenberg–Marquardt algorithm to the residual force $f(x)$. Use $x^{(1)} = (0, 0)$ as starting point. (Note that it is important to start at a point where $T_1 > 0$ and $T_2 > 0$, because otherwise the derivative matrix $Df(x^{(1)})$ is zero, and the Levenberg–Marquardt update gives $x^{(2)} = x^{(1)}$.) Plot the components of the mass position and the residual force versus iterations.

Solution. We write the residual force as

$$f(x) = F^{\text{grav}} + g_1(x)(a_1 - x) + g_2(x)(a_2 - x)$$

where

$$g_1(x) = k \max\{\frac{1}{L_1} - \frac{1}{\|a_1 - x\|}, 0\}, \qquad g_2(x) = k \max\{\frac{1}{L_2} - \frac{1}{\|a_2 - x\|}, 0\}.$$

The derivative matrix is

$$Df(x) = -g_1(x)I + (a_1 - x)\nabla g_1(x)^T - g_2(x)I + (a_2 - x)\nabla g_2(x)^T$$

where

$$\nabla g_1(x) = \begin{cases} 0 & \|a_1 - x\| < L_1 \\ \frac{-k}{\|a_1 - x\|^{3/2}}(a_1 - x) & \|a_1 - x\| > L_1 \end{cases}$$

and

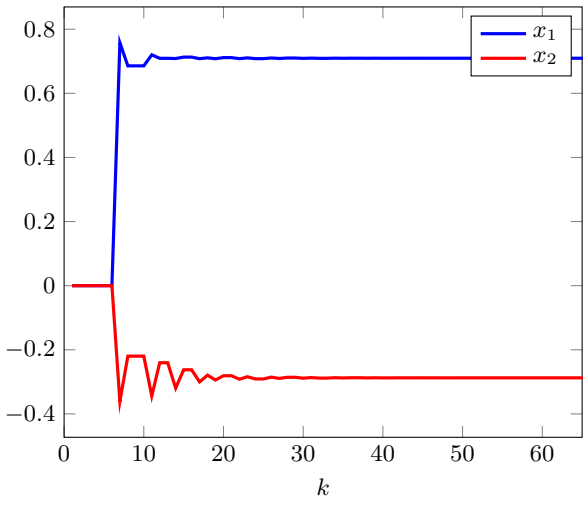
$$\nabla g_2(x) = \begin{cases} 0 & \|a_2 - x\| < L_2 \\ \frac{-k}{\|a_2 - x\|^{3/2}}(a_2 - x) & \|a_2 - x\| > L_2. \end{cases}$$

Note that g_1 is not differentiable at x if $\|a_1 - x\| = L_1$, and similarly for g_2.

We start the Levenberg–Marquardt algorithm at $x^{(1)} = (0, 0)$ and $\lambda^{(1)} = 1$, and terminate when $\|2Df(x^{(k)})^T f(x^{(k)})\| < 10^{-5}$. It converges to

$$x = (0.7095, -0.2875), \qquad \|a_1 - x\| = 3.237, \qquad \|a_2 - x\| = 2.140.$$

The figure shows the components of $x^{(k)}$ and $f(x^{(k)})$.



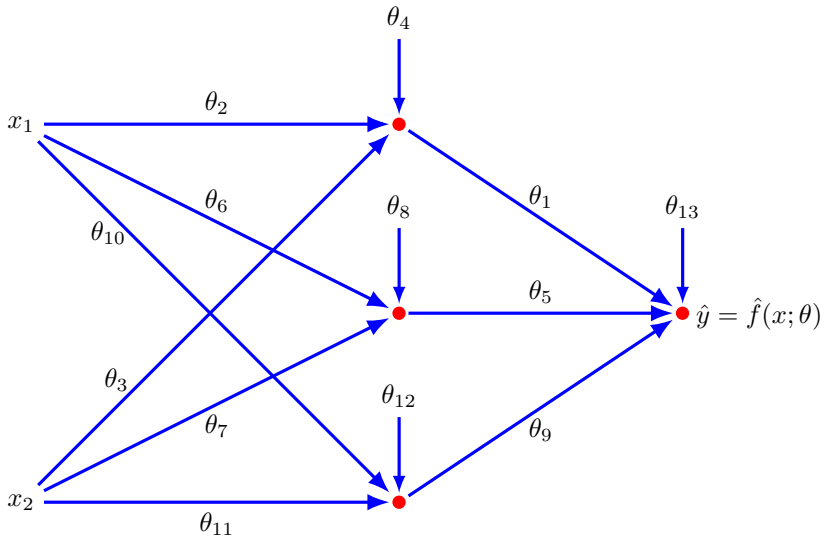
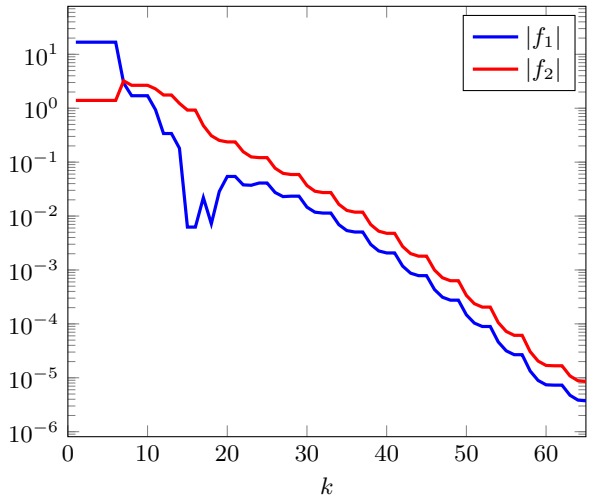


Figure 18.1 Signal flow graph of a simple neural network.



18.6 *Fitting a simple neural network model.* A neural network is a widely used model of the form $\hat{y} = \hat{f}(x;\theta)$, where the n-vector x is the feature vector and the p-vector θ is the model parameter. In a neural network model, the function $\hat{f}$ is *not* an affine function of the parameter vector θ. In this exercise we consider a very simple neural network, with two layers, three internal nodes, and two inputs (*i.e.*, $n = 2$). This model has $p = 13$ parameters, and is given by

$$\begin{aligned}\hat{f}(x;\theta) &= \theta_1\phi(\theta_2 x_1 + \theta_3 x_2 + \theta_4) + \theta_5\phi(\theta_6 x_1 + \theta_7 x_2 + \theta_8) \\ &\quad + \theta_9\phi(\theta_{10} x_1 + \theta_{11} x_2 + \theta_{12}) + \theta_{13}\end{aligned}$$

where $\phi : \mathbf{R} \to \mathbf{R}$ is the sigmoid function defined in (18.16). This function is shown as a *signal flow graph* in figure 18.1. In this graph each edge from an input to an internal

node, or from an internal node to the output node, corresponds to multiplication by one of the parameters. At each node (shown as the small filled circles) the incoming values and the constant offset are added together, then passed through the sigmoid function, to become the outgoing edge value.

Fitting such a model to a data set consisting of the n-vectors $x^{(1)}, \ldots, x^{(N)}$ and the associated scalar outcomes $y^{(1)}, \ldots, y^{(N)}$ by minimizing the sum of the squares of the residuals is a nonlinear least squares problem with objective (18.4).

(a) Derive an expression for $\nabla_\theta \hat{f}(x;\theta)$. Your expression can use ϕ and ϕ', the sigmoid function and its derivative. (You do not need to express these in terms of exponentials.)

(b) Derive an expression for the derivative matrix $Dr(\theta)$, where $r : \mathbf{R}^p \to \mathbf{R}^N$ is the vector of model fitting residuals,

$$r(\theta)_i = \hat{f}(x^{(i)};\theta) - y^{(i)}, \quad i = 1, \ldots, N.$$

Your expression can use the gradient found in part (a).

(c) Try fitting this neural network to the function $g(x_1, x_2) = x_1 x_2$. First generate $N = 200$ random points $x^{(i)}$ and take $y^{(i)} = (x^{(i)})_1 (x^{(i)})_2$ for $i = 1, \ldots, 200$. Use the Levenberg–Marquardt algorithm to try to minimize

$$f(\theta) = \|r(\theta)\|^2 + \gamma \|\theta\|^2$$

with $\gamma = 10^{-5}$. Plot the value of f and the norm of its gradient versus iteration. Report the RMS fitting error achieved by the neural network model. Experiment with choosing different starting points to see the effect on the final model found.

(d) Fit the same data set with a (linear) regression model $\hat{f}^{\mathrm{lin}}(x;\beta,v) = x^T\beta + v$ and report the RMS fitting error achieved. (You can add regularization in your fitting, but it won't improve the results.) Compare the RMS fitting error with the neural network model RMS fitting error from part (c).

Remarks. Neural networks used in practice employ many more regressors, layers, and internal modes. Specialized methods and software are used to minimize the fitting objective, and evaluate the required gradients and derivatives.

Solution.

(a) The gradient is

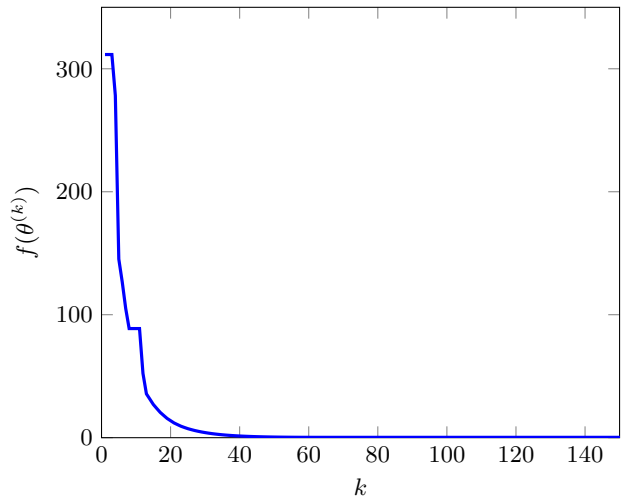
$$\nabla_\theta \hat{f}(x;\theta) = \begin{bmatrix} \phi(\theta_2 x_1 + \theta_3 x_2 + \theta_4) \\ \theta_1 x_1 \phi'(\theta_2 x_1 + \theta_3 x_2 + \theta_4) \\ \theta_1 x_2 \phi'(\theta_2 x_1 + \theta_3 x_2 + \theta_4) \\ \theta_1 \phi'(\theta_2 x_1 + \theta_3 x_2 + \theta_4) \\ \phi(\theta_6 x_1 + \theta_7 x_2 + \theta_8) \\ \theta_6 x_1 \phi'(\theta_6 x_1 + \theta_7 x_2 + \theta_8) \\ \theta_7 x_2 \phi'(\theta_6 x_1 + \theta_7 x_2 + \theta_8) \\ \theta_8 \phi'(\theta_6 x_1 + \theta_7 x_2 + \theta_8) \\ \phi(\theta_{10} x_1 + \theta_{11} x_2 + \theta_{12}) \\ \theta_{10} x_1 \phi'(\theta_{10} x_1 + \theta_{11} x_2 + \theta_{12}) \\ \theta_{10} x_2 \phi'(\theta_{10} x_1 + \theta_{11} x_2 + \theta_{12}) \\ \theta_{10} \phi'(\theta_{10} x_1 + \theta_{11} x_2 + \theta_{12}) \end{bmatrix}$$

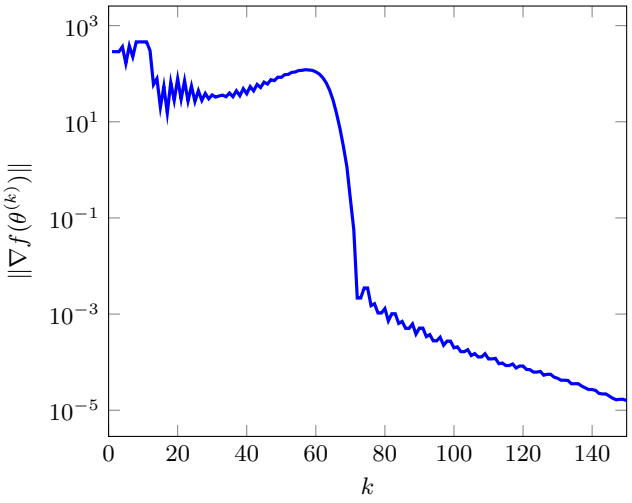
where $\phi'(u) = 4/(e^u + e^{-u})^2$.

(b) The ith row of $Dr(\theta)$ is the transpose of the $\nabla_\theta \hat{f}(x^{(i)};\theta)$.

(c) We choose the 200 points from a standard normal distribution. First we work out the derivative matrix for the objective function f. Its ith row is XXX.

The first two figures show the value of the objective function $f(\theta) = \|r(\theta)\|^2 + \gamma\|\theta\|^2$ and the norm of its gradient versus iteration number. The algorithm was started with randomly chosen parameters.





The next figure show the contour lines of $\hat{f}(x;\theta)$ (left) and the function x_1x_2 (right);

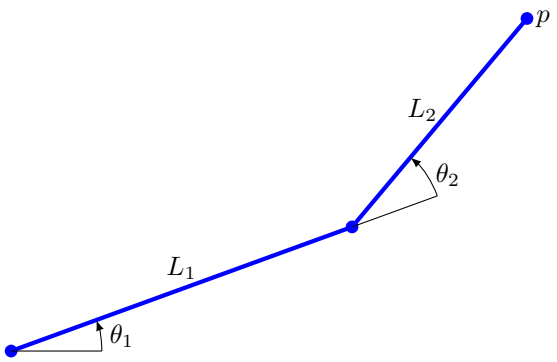
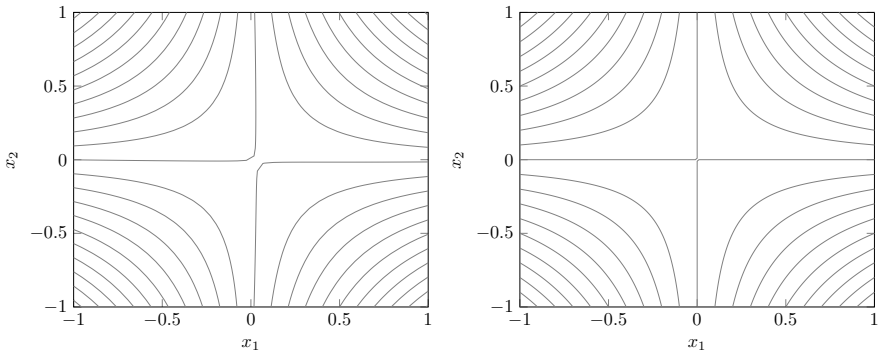
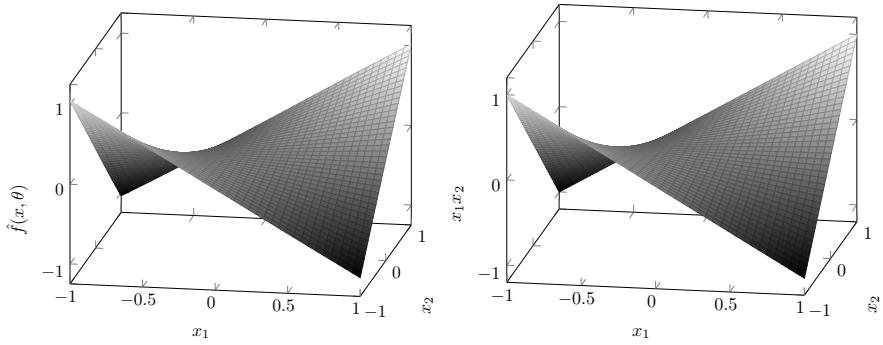


Figure 18.2 Two-link robot manipulator in a plane.



The last figure show the graph of $\hat{f}(x;\theta)$ (left) and the function x_1x_2 (right);



(d) The RMS fitting error for the neural network is 0.027. The RMS fitting error for a linear regression model (computed without regularization) is is 1.12.

18.7 *Robot manipulator.* Figure 18.2 shows a two-link robot manipulator in a plane. The robot manipulator endpoint is at the position

$$p = L_1 \begin{bmatrix} \cos\theta_1 \\ \sin\theta_1 \end{bmatrix} + L_2 \begin{bmatrix} \cos(\theta_1+\theta_2) \\ \sin(\theta_1+\theta_2) \end{bmatrix},$$

where L_1 and L_2 are the lengths of the first and second links, θ_1 is the first joint angle, and θ_2 is second joint angle. We will assume that $L_2 < L_1$, *i.e.*, the second link is shorter than

the first. We are given a desired endpoint position p^{des}, and seek joint angles $\theta = (\theta_1, \theta_2)$ for which $p = p^{\text{des}}$.
This problem can be solved analytically. We find θ_2 using the equation

$$\begin{aligned} \|p^{\text{des}}\|^2 &= (L_1 + L_2\cos\theta_2)^2 + (L_2\sin\theta_2)^2 \\ &= L_1^2 + L_2^2 + 2L_1L_2\cos\theta_2. \end{aligned}$$

When $L_1 - L_2 < \|p^{\text{des}}\| < L_1 + L_2$, there are two choices of θ_2 (one positive and one negative). For each solution θ_2 we can then find θ_1 using

$$\begin{aligned} p^{\text{des}} &= L_1 \begin{bmatrix} \cos\theta_1 \\ \sin\theta_1 \end{bmatrix} + L_2 \begin{bmatrix} \cos(\theta_1+\theta_2) \\ \sin(\theta_1+\theta_2) \end{bmatrix} \\ &= \begin{bmatrix} L_1 + L_2\cos\theta_2 & -L_2\sin\theta_2 \\ L_2\sin\theta_2 & L_1 + L_2\cos\theta_2 \end{bmatrix} \begin{bmatrix} \cos\theta_1 \\ \sin\theta_1 \end{bmatrix}. \end{aligned}$$

In this exercise you will use the Levenberg–Marquardt algorithm to find joint angles, by minimizing $\|p - p^{\text{des}}\|^2$.

(a) Identify the function $f(\theta)$ in the nonlinear least squares problem, and give its derivative $Df(\theta)$.

(b) Implement the Levenberg–Marquardt algorithm to solve the nonlinear least squares problem. Try your implementation on a robot with $L_1 = 2$, $L_2 = 1$, and the desired endpoints

$$(1.0, 0.5), \quad (-2.0, 1.0), \quad (-0.2, 3.1).$$

For each endpoint, plot the cost function $\|f(\theta^{(k)})\|^2$ versus iteration number k.
Note that the norm of the last endpoint exceeds $L_1 + L_2 = 3$, so there are no joint angles for which $p = p^{\text{des}}$. Explain the angles your algorithm finds in this case.

Solution.

(a) We take

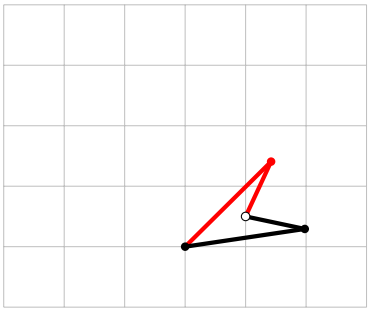
$$f(\theta) = p - p^{\text{des}} = L_1 \begin{bmatrix} \cos\theta_1 \\ \sin\theta_1 \end{bmatrix} + L_2 \begin{bmatrix} \cos(\theta_1+\theta_2) \\ \sin(\theta_1+\theta_2) \end{bmatrix} - p^{\text{des}}.$$

The derivative is

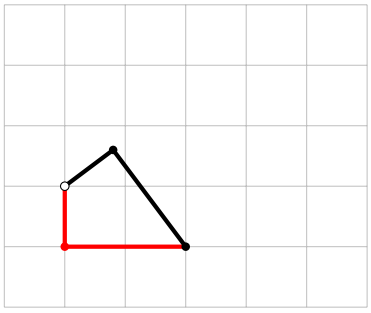
$$Df(\theta) = \begin{bmatrix} -L_1\sin\theta_1 - L_2\sin(\theta_1+\theta_2) & -L_2\sin(\theta_1+\theta_2) \\ L_1\cos\theta_1 + L_2\cos(\theta_1+\theta_2) & L_2\cos(\theta_1+\theta_2) \end{bmatrix}.$$

(b) The first two figures show the solutions for $p^{\text{des}} = (1.0, 0.5)$ (left) and $p^{\text{des}} = (-2.0, 1.0)$ (right). In each case there are two solutions, shown in red and black. The desired endpoint is shown as an open circle.

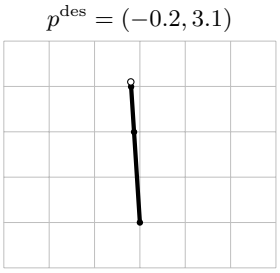
$p^{\text{des}} = (1.0, 0.5)$



$p^{\text{des}} = (-2.0, 1.0)$



The next figure shows the solution for endpoint $(-0.2, 3.1)$.



For $p^{\text{des}} = (1, 0.5)$ the two solutions are

$$(\theta_1, \theta_2) = (8.43^\circ, 159.6^\circ), \qquad (\theta_1, \theta_2) = (44.7^\circ, -159.6^\circ).$$

When started at $(\theta_1, \theta_2) = 0$, the Levenberg–Marquardt algorithm converges to the first solution in about 15 iterations. For $p^{\text{des}} = (-2, 1)$, the solutions are

$$(\theta_1, \theta_2) = (126.9^\circ, 90^\circ), \qquad (\theta_1, \theta_2) = (180^\circ, -90^\circ).$$

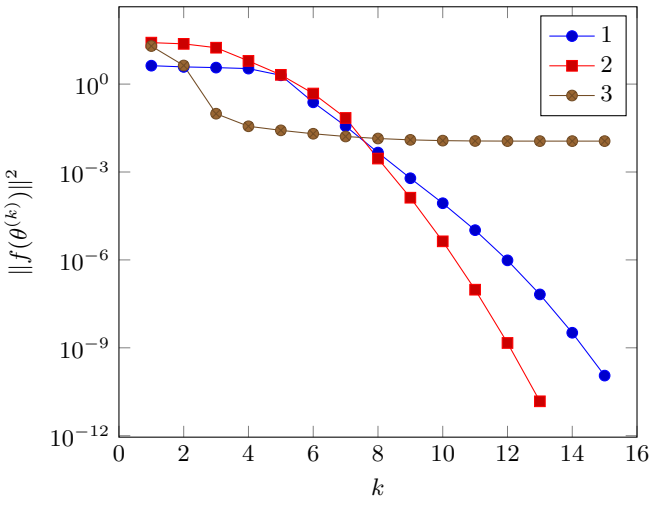
When started at $(\theta_1, \theta_2) = 0$, the Levenberg–Marquardt algorithm converges to the first solution in about 13 iterations.

In the third problem, the Levenberg–Marquardt iteration converges to

$$(\theta_1, \theta_2) = (93.7^\circ, 0).$$

The two links form a straight line, so the end point is the point at a distance $L_1 + L_2$ from the origin closest to p^{des}.

The convergence plots for the three endpoints are shown below.



18.8 *Fitting an ellipse to points in a plane.* An ellipse in a plane can be described as the set of points

$$\hat{f}(t;\theta) = \begin{bmatrix} c_1 + r\cos(\alpha + t) + \delta\cos(\alpha - t) \\ c_2 + r\sin(\alpha + t) + \delta\sin(\alpha - t) \end{bmatrix},$$

where t ranges from 0 to 2π. The vector $\theta = (c_1, c_2, r, \delta, \alpha)$ contains five parameters, with geometrical meanings illustrated in figure 18.3. We consider the problem of fitting an

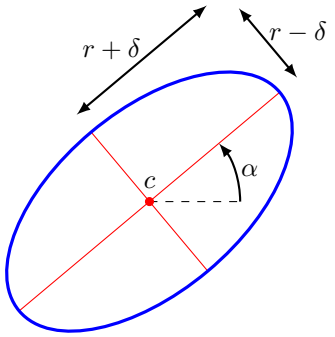


Figure 18.3 Ellipse with center (c_1, c_2), and radii $r+\delta$ and $r-\delta$. The largest semi-axis makes an angle α with respect to horizontal.

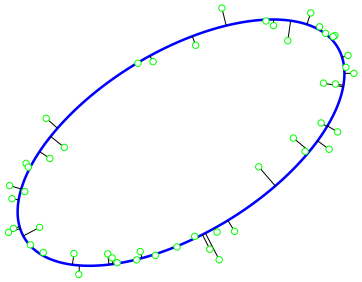


Figure 18.4 Ellipse fit to 50 points in a plane.

ellipse to N points $x^{(1)}, \ldots, x^{(N)}$ in a plane, as shown in figure 18.4. The circles show the N points. The short lines connect each point to the nearest point on the ellipse. We will fit the ellipse by minimizing the sum of the squared distances of the N points to the ellipse.

(a) The squared distance of a data point $x^{(i)}$ to the ellipse is the minimum of $\|\hat{f}(t^{(i)};\theta) - x^{(i)}\|^2$ over the scalar $t^{(i)}$. Minimizing the sum of the squared distances of the data points $x^{(1)}, \ldots, x^{(N)}$ to the ellipse is therefore equivalent to minimizing

$$\sum_{i=1}^{N} \|\hat{f}(t^{(i)};\theta) - x^{(i)}\|^2$$

over $t^{(1)}, \ldots, t^{(N)}$ and θ. Formulate this as a nonlinear least squares problem. Give expressions for the derivatives of the residuals.

(b) Use the Levenberg–Marquardt algorithm to fit an ellipse to the 10 points:

$$(0.5, 1.5), \quad (-0.3, 0.6), \quad (1.0, 1.8), \quad (-0.4, 0.2), \quad (0.2, 1.3)$$

$$(0.7, 0.1), \quad (2.3, 0.8), \quad (1.4, 0.5), \quad (0.0, 0.2), \quad (2.4, 1.7).$$

To select a starting point, you can choose parameters θ that describe a circle with radius one and center at the average of the data points, and initial values of $t^{(i)}$ that minimize the objective function for those initial values of θ.

Solution.

(a) We use $N + 5$ variables

$$x = (c_1, c_2, r, \delta, \alpha, t_1, \ldots, t_N)$$

and a residual vector $f(x)$ with $2N$ components:

$$f_i(x) = c_1 + r\cos(\alpha + t^{(i)}) + \delta\cos(\alpha - t^{(i)}) - x_{i1}, \quad i = 1, \ldots, N,$$

and

$$f_{N+i}(x) = c_2 + r\sin(\alpha + t^{(i)}) + \delta\sin(\alpha - t^{(i)}) - x_{i2}, \quad i = 1, \ldots, N,$$

Then $Df(x)$ is the $2N \times (N + 5)$ matrix

$$\begin{bmatrix} \mathbf{1} & 0 & u_1 & u_2 & -rv_1 - \delta v_2 & \mathbf{diag}(-rv_1 + \delta v_2) \\ 0 & \mathbf{1} & v_1 & v_2 & ru_1 + \delta u_2 & \mathbf{diag}(ru_1 - \delta u_2) \end{bmatrix}.$$

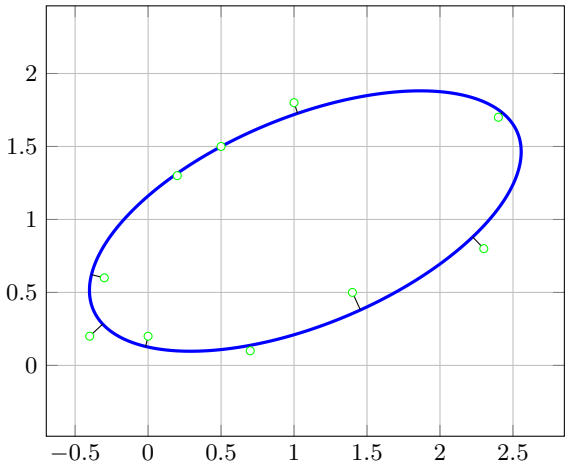
where u_1, u_2, v_1, and v_2 are N-vectors with elements

$$u_{1i} = \cos(\alpha + t^{(i)}), \quad u_{2i} = \cos(\alpha - t^{(i)}), \quad v_{1i} = \sin(\alpha + t^{(i)}), \quad v_{2i} = \sin(\alpha - t^{(i)}),$$

for $i = 1, \ldots, N$.

(b) The Levenberg–Marquardt method converges to the ellipse defined by

$$c_1 = 1.0773, \qquad c_2 = 0.9891, \qquad r = 1.1421, \qquad \delta = 0.4327, \qquad \alpha = 0.3949.$$



Chapter 19

Constrained nonlinear least squares

Exercises

19.1 *Projection on a curve.* We consider a constrained nonlinear least squares problem with three variables $x = (x_1, x_2, x_3)$ and two equations:

$$\begin{array}{ll} \text{minimize} & (x_1-1)^2 + (x_2-1)^2 + (x_3-1)^2 \\ \text{subject to} & x_1^2 + 0.5x_2^2 + x_3^2 - 1 = 0 \\ & 0.8x_1^2 + 2.5x_2^2 + x_3^2 + 2x_1x_3 - x_1 - x_2 - x_3 - 1 = 0. \end{array}$$

The solution is the point closest to $(1, 1, 1)$ on the nonlinear curve defined by the two equations.

(a) Solve the problem using the augmented Lagrangian method. You can start the algorithm at $x^{(1)} = 0$, $z^{(1)} = 0$, $\mu^{(1)} = 1$, and start each run of the Levenberg–Marquardt method with $\lambda^{(1)} = 1$. Stop the augmented Lagrangian method when the feasibility residual $\|g(x^{(k)})\|$ and the optimality condition residual

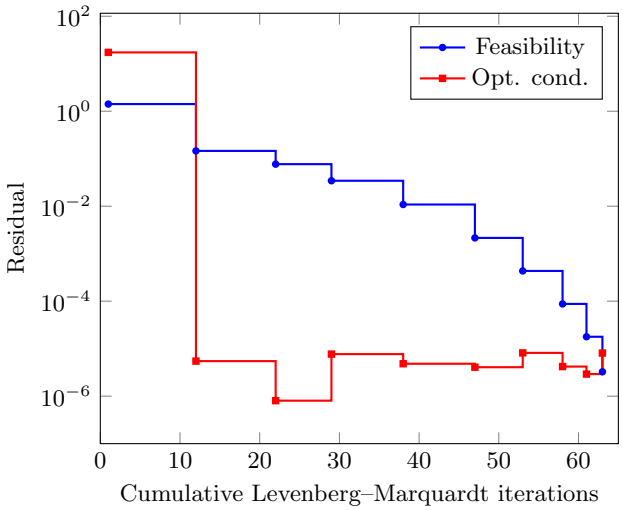
$$\|2Df(x^{(k)})^T f(x^{(k)}) + Dg(x^{(k)})^T z^{(k)}\|$$

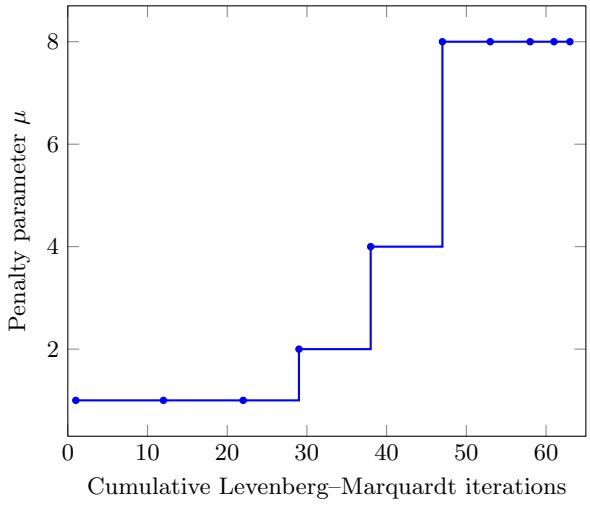
are less than 10^{-5}. Make a plot of the two residuals and of the penalty parameter μ versus the cumulative number of Levenberg–Marquardt iterations.

(b) Solve the problem using the penalty method, started at $x^{(1)} = 0$ and $\mu^{(1)} = 1$, and with the same stopping condition. Compare the convergence and the value of the penalty parameter with the results for the augmented Lagrangian method in part (a).

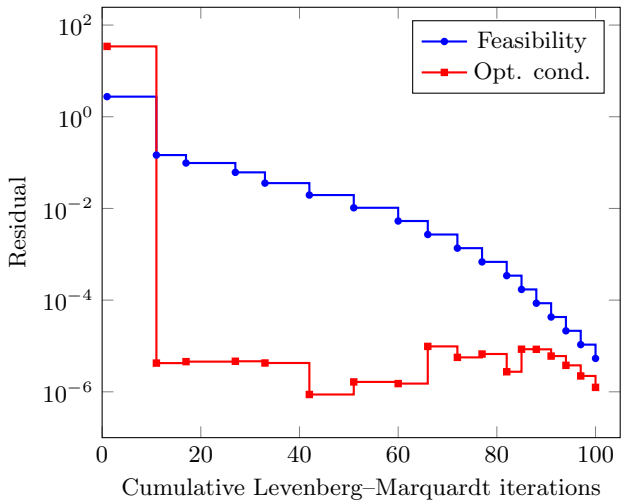
Solution

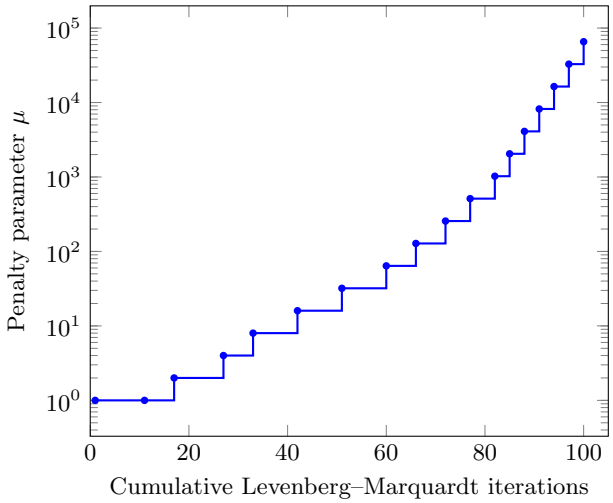
(a) The algorithm finds the point $x = (0.568, 0.833, 0.575)$ in 9 iterations and a total of 63 Levenberg–Marquardt iterations. The plots are shown below.





(b) The penalty method finds the same point in 17 iterations and a total of 100 Levenberg–Marquardt iterations.





19.2 *Portfolio optimization with downside risk.* In standard portfolio optimization (as described in §17.1) we choose the weight vector w to achieve a given target mean return, and to minimize deviations from the target return value (*i.e.*, the risk). This leads to the constrained linear least squares problem (17.2). One criticism of this formulation is that it treats portfolio returns that exceed our target value the same as returns that fall short of our target value, whereas in fact we should be delighted to have a return that exceeds our target value. To address this deficiency in the formulation, researchers have defined the *downside risk* of a portfolio return time series T-vector r, which is sensitive only to portfolio returns that fall short of our target value ρ^{tar}. The downside risk of a portfolio return time series (T-vector) r is given by

$$D = \frac{1}{T}\sum_{t=1}^{T}\left(\max\left\{\rho^{\text{tar}} - r_t, 0\right\}\right)^2.$$

The quantity $\max\{\rho^{\text{tar}} - r_t, 0\}$ is the shortfall, *i.e.*, the amount by which the return in period t falls short of the target; it is zero when the return exceeds the target value. The downside risk is the mean square value of the return shortfall.

(a) Formulate the portfolio optimization problem, using downside risk in place of the usual risk, as a constrained nonlinear least squares problem. Be sure to explain what the functions f and g are.

(b) Since the function g is affine (if you formulate the problem correctly), you can use the Levenberg–Marquardt algorithm, modified to handle linear equality constraints, to approximately solve the problem. (See page 420.) Find an expression for $Df(x^{(k)})$. You can ignore the fact that the function f is not differentiable at some points.

(c) Implement the Levenberg–Marquardt algorithm to find weights that minimize downside risk for a given target annualized return. A very reasonable starting point is the solution of the standard portfolio optimization problem with the same target return. Check your implementation with some simulated or real return data (available online). Compare the weights, and the risk and downside risk, for the minimum risk and the minimum downside risk portfolios.

Solution.

(a) We formulate the problem as

$$\begin{array}{ll} \text{minimize} & \sum_{t=1}^{T}(\max\{\rho^{\text{tar}} - (Rw)_t, 0\})^2 \\ \text{subject to} & \mathbf{1}^T w = 1 \\ & \mu^T w = \rho^{\text{tar}} \end{array}$$

where $\mu = (1/T)R^T\mathbf{1}$. This is a constrained least squares problem with $m = T$, $p = 2$,

$$f(w) = \begin{bmatrix} \max\{\rho^{\text{tar}} - (Rw)_1, 0\} \\ \vdots \\ \max\{\rho^{\text{tar}} - (Rw)_T, 0\} \end{bmatrix}, \qquad g(w) = \begin{bmatrix} \mathbf{1}^T w - 1 \\ \mu^T w - \rho^{\text{tar}} \end{bmatrix}.$$

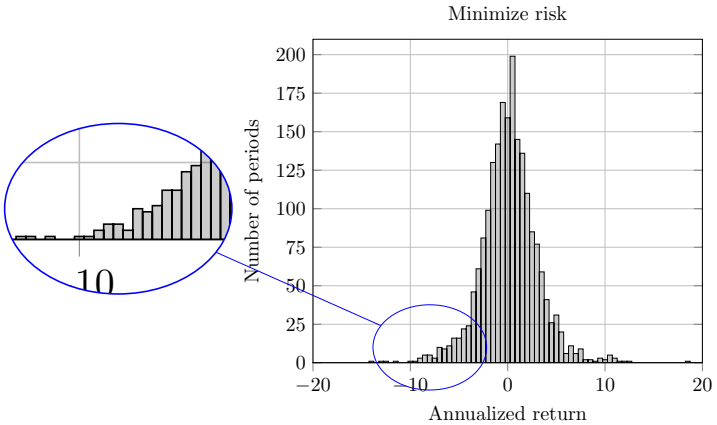
(b) The derivative matrix $Df(w)$ is

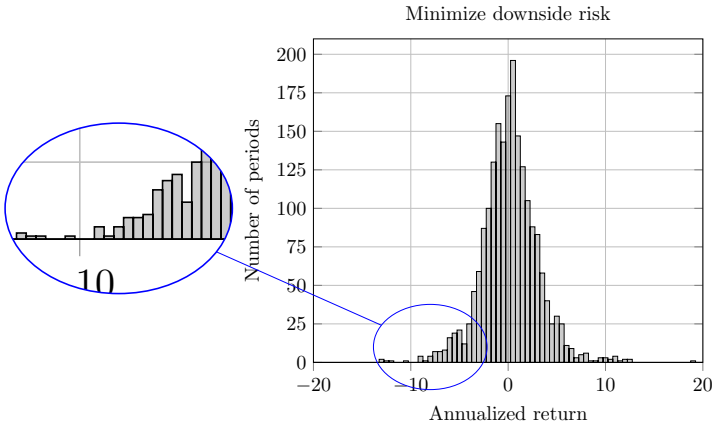
$$Df(w) = -\,\mathbf{diag}(\alpha_1, \ldots, \alpha_T)R, \qquad \alpha_k = \begin{cases} 0 & (Rw)_k > \rho^{\text{tar}} \\ 1 & (Rw)_k < \rho^{\text{tar}}. \end{cases}$$

(c) We use the return data of the example in §17.1 ($n = 20$ and $T = 2000$), and $\rho^{\text{tar}} = 0.20/250$. We first compute the portfolio w_{ls} that minimizes risk for return value ρ^{tar}, by solving the constrained least squares problem (17.2). The downside risk for this portfolio is $D(w_{\text{ls}}) = 6.50\,10^{-5}$, where

$$D(w) = \frac{1}{T}\sum_{t=1}^{T}\left(\max\{\rho^{\text{tar}} - (Rw)_t, 0\}\right)^2.$$

This corresponds to an annualized value $\sqrt{250D(w_{\text{ls}})} = 12.75\%$. We use this portfolio as starting point for the Levenberg–Marquardt algorithm. The algorithm converges to a portfolio with downside risk $D(w) = 6.45\,10^{-4}$. This corresponds to an annualized value $\sqrt{250D(w)} = 12.70\%$. The distribution of the annualized returns $250(Rw)_t$ for the two portfolios are as follows.

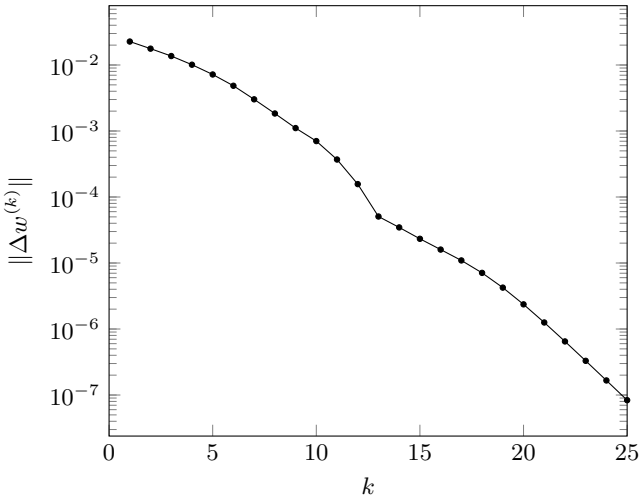




In the implementation of the Levenberg–Marquardt algorithm we define the tentative iterate at iteration k as the solution of the constrained least squares problem

$$\begin{array}{ll} \text{minimize} & \|f(w^{(k)}) + Df(w^{(k)})(w - w^{(k)}\|^2 + \lambda\|w - w^{(k)}\|^2 \\ \text{subject to} & g(w) = 0. \end{array}$$

In the following figure Δw is the defined as $w - w^{(k)}$, where w is this tentative iterate.



19.3 *Boolean least squares.* The *Boolean least squares problem* is a special case of the constrained nonlinear least squares problem (19.1), with the form

$$\begin{array}{ll} \text{minimize} & \|Ax - b\|^2 \\ \text{subject to} & x_i^2 = 1, \quad i = 1, \ldots, n, \end{array}$$

where the n-vector x is the variable to be chosen, and the $m \times n$ matrix A and the m-vector b are the (given) problem data. The constraints require that each entry of x is either -1 or $+1$, *i.e.*, x is a Boolean vector. Since each entry can take one of two values, there are 2^n feasible values for the vector x. The Boolean least squares problem arises in many applications.

One simple method for solving the Boolean least squares problem, sometimes called the *brute force method*, is to evaluate the objective function $\|Ax - b\|^2$ for each of the 2^n possible values, and choose one that has the least value. This method is not practical for n larger than 30 or so. There are many heuristic methods that are much faster to carry out than the brute force method, and approximately solve it, *i.e.*, find an x for which the objective is small, if not the smallest possible value over all 2^n feasible values of x. One such heuristic is the augmented Lagrangian algorithm 19.2.

(a) Work out the details of the update step in the Levenberg–Marquardt algorithm used in each iteration of the augmented Lagrangian algorithm, for the Boolean least squares problem.

(b) Implement the augmented Lagrangian algorithm for the Boolean least squares problem. You can choose the starting point $x^{(1)}$ as the minimizer of $\|Ax - b\|^2$. At each iteration, you can obtain a feasible point $\tilde{x}^{(k)}$ by rounding the entries of $x^{(k)}$ to the values ± 1, *i.e.*, $\tilde{x}^{(k)} = \mathbf{sign}(x^{(k)})$. You should evaluate and plot the objective value of these feasible points, *i.e.*, $\|A\tilde{x}^{(k)} - b\|^2$. Your implementation can return the best rounded value found during the iterations. Try your method on some small problems, with $n = m = 10$ (say), for which you can find the actual solution by the brute force method. Try it on much larger problems, with $n = m = 500$ (say), for which the brute force method is not practical.

Solution.

(a) In this problem, $f(x) = Ax - b$ and $Df(x) = A$, and

$$g(x) = (x_1^2 - 1, \ldots, x_n^2 - 1), \qquad Dg(x) = 2\,\mathbf{diag}(x).$$

In the kth iteration of the augmented Lagrangian method, we solve the nonlinear least squares problem

$$\text{minimize} \quad \|Ax - b\|^2 + \mu^{(k)}\|g(x) + z^{(k)}/(2\mu^{(k)})\|^2.$$

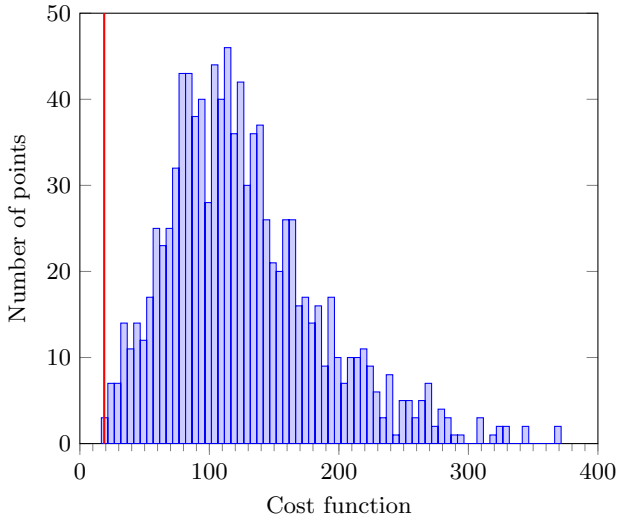
In the ith iteration of the Levenberg–Marquardt method applied to this problem, we minimize

$$\|Ax - b\|^2 + \mu^{(k)}\|Dg(x^{(i)})(x - x^{(i)}) + g(x^{(i)}) + \frac{1}{2\mu^{(k)}}z^{(k)}\|^2 + \lambda^{(i)}\|x - x^{(i)}\|^2$$

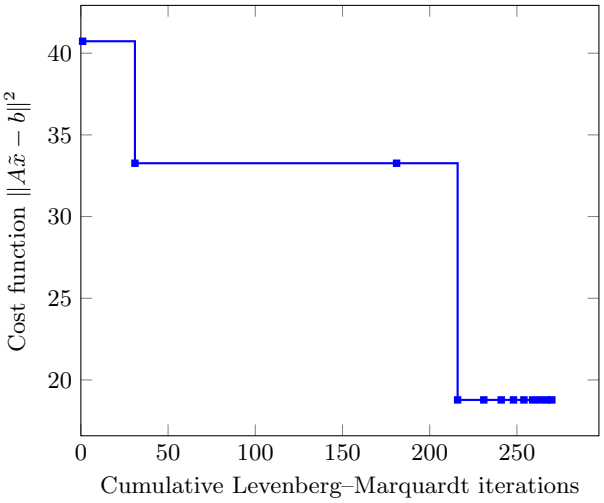
to compute $x^{(i+1)}$. If we solve for the step $v = x^{(i+1)} - x^{(i)}$, this linear least squares problem can be written as minimizing

$$\left\| \begin{bmatrix} A \\ 2\sqrt{\mu^{(k)}}\,\mathbf{diag}(x^{(i)}) \\ \sqrt{\lambda^{(i)}}I \end{bmatrix} v + \begin{bmatrix} Ax^{(i)} - b \\ \sqrt{\mu^{(k)}}(g(x^{(i)}) + 1/(2\mu^{(k)})z^{(k)}) \\ 0 \end{bmatrix} \right\|^2.$$

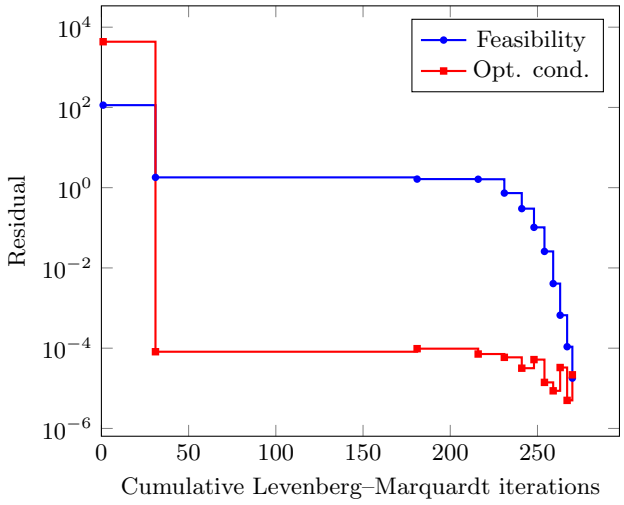
(b) We first generate some examples of size $n = m = 10$, with the entries of A and b from a standard normal distribution. The plots show the results of a typical example. The first plot is the histogram of $\|Ax - b\|^2$ at the $2^n = 1024$ possible values of x. The optimal value is 16.586. If we round the solution returned by the augmented Lagrangian method we obtain a vector with $\|A\tilde{x} - b\|^2 = 18.778$ (indicated with a vertical line on the histogram).



The next figure shows the cost function at the rounded solution versus the cumulative number of Levenberg–Marquardt iterations. We note that the cost function remains constant after three iterations in the augmented Lagrangian method.

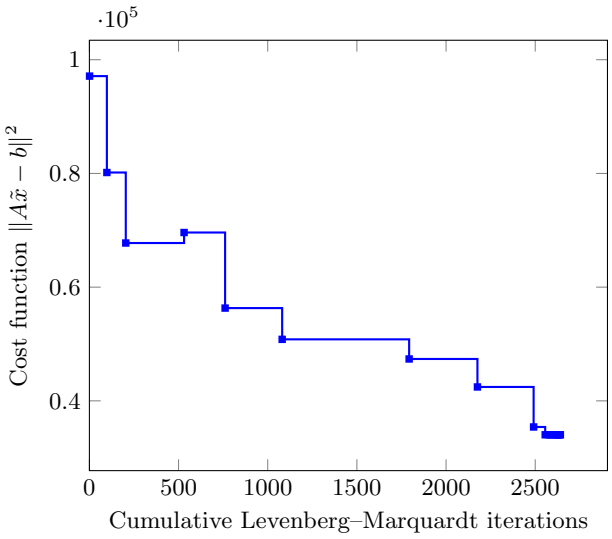


The next figure shows the feasibility and optimality condition error versus the cumulative number of Levenberg–Marquardt iterations.



In this implementation we followed the outline of algorithms 19.2 and 18.3, with the same parameter values (and $\lambda^{(1)} = 1$ in the Levenberg–Marquardt algorithm).

The next two figures show the convergence for a larger example, with $n = m = 500$.



The next figure shows the feasibility and optimality condition error versus the cumulative number of Levenberg–Marquardt iterations.