Additional Exercises for

Introduction to Applied Linear Algebra: Vectors, Matrices, and Least Squares

Stephen Boyd Lieven Vandenberghe

(with additions by John Duchi)

September 2024

This is a collection of additional exercises for the book *Introduction to Applied Linear Algebra: Vectors, Matrices, and Least Squares* (Cambridge 2018), by Stephen Boyd and Lieven Vandenberghe. They are used in ENGR 108 (Stanford) and EE 133A (UCLA). Unlike the exercises in the book, some of these specifically include numerical exercises using Julia. The first 19 sections follow the book chapters. We update these exercises whenever we create new problems, *e.g.*, as exam or homework problems in our courses.

Stephen Boyd and Lieven Vandenberghe

Contents

1 Vectors

1.1 *Julia timing test.* Determine how much time it takes for your computer to compute the inner product of two vectors of length 10^8 (100 million), and use this to estimate (very crudely) how many Gflops/sec your computer can carry out. The following code generates two (random) vectors of length 10^8, and times the evaluation of the inner product. (You might run it a few times; the first time might be a bit slower.)

```
a = randn(10^8);
b = randn(10^8);
@time s=a'*b
```

How long would it take a person to carry this out, assuming the person can carry out a floating point operation every 10 seconds for 8 hours each day?

1.2 *Creating vectors in Julia.* In each of the parts below, use Julia to create the described vector a. In each case, check that $a^T x$ gives the correct result, for a random vector x.

(a) $a^T x$ extracts (is equal to) the 5th entry of the 10-vector x.

(b) $a^T x$ is the weighted average of a 3-vector x, assigning weights 0.3 to the first component, 0.4 to the second, and 0.3 to the third.

(c) $a^T x$ (with x a 22-vector) is the sum of x_i for i a multiple of a 4, minus the sum of x_i for i a multiple of 7.

(d) $a^T x$ (with x an 11-vector) is the average of the middle five entries of x, *i.e.*, entries 4 to 8.

1.3 *Vector expressions.* Suppose $a = (4, -2, 1)$. Determine whether each expression below is valid notation. If it is, give its value.

(a) (a, a)

(b) $a - [4, -2, 0]$

(c) $a - \begin{bmatrix} 4 \\ -2 \\ 3 \end{bmatrix}$

(d) $a + (2, 6)$

1.4 The 7-vector c represents the daily earnings of a company over one week, with c_1 the earnings on Sunday, c_2 the earnings on Monday, and so on, with c_7 the earnings on Saturday. (Negative entries in the earnings vector mean a loss on that day.)

Express the following quantities in the form $a^T c$. In each case, give a (which can be different for the different quantities, of course).

(a) Wednesday's earnings.

(b) The total earnings over the week.

(c) The average weekend earnings.

(d) The average weekday earnings.

(e) The difference between the average weekend earnings and the average weekday earnings.

1.5 *When an inner product or scalar-vector product is zero.* Given n-vectors a, b, x, and scalar α. Select one choice in each subproblem below.

(a) If $a^Tb = 0$, then

(i) at least one of a or b must be 0

(ii) both a and b must be 0

(iii) both a and b can be nonzero

(b) If $\alpha x = 0$, then

(i) one of α or x must be 0

(ii) both α and x must be 0

(iii) both α and x can be nonzero

1.6 A particular computer can compute the inner product of two 10^6-vectors in around 0.001 second. The same computer can compute the inner product of two 10^7-vectors in (approximately) how long?

1.7 The regression model $\hat{y} = x^T\beta + v$ predicts the life span (age at death) of a person in some population, where the feature vector x encodes various attributes of the person. Assuming the model fits actual life span data reasonably well (although of course not very accurately for any particular individual) what would you guess about β_3, if $x_3 = 1$ means the person is a smoker, and $x_3 = 0$ means the person is not a smoker? For example, do you think that β_3 is positive? Negative? Feel free to guess a plausible value for β_3.

1.8 *Some basic vector operations.* Suppose $w = (0, 2, -1)$ and $z = (-3, 2)$. Determine whether each expression below uses valid (mathematical) notation, and if it is valid, give its value.

(a) $w_{2:3} + z$.

(b) $w - 2z$.

(c) $[w; -z]$.

(d) $\begin{bmatrix} z \\ w \\ 2z \end{bmatrix} + \mathbf{1}$.

(e) $w^T(z, 3)$.

1.9 *Prices, bill of materials, and a budget.* To produce one unit of some product requires specific quantities of n different resources, denoted $q_1, \ldots, q_n$, each with its own units. (The vector q is called the bill of materials for the product.) To produce Q units of the product, the resource requirements are $Qq_1, \ldots, Qq_n$. The prices per unit of the resources are given by $p_1, \ldots, p_n$. (The quantities and prices are positive.) The total cost is the sum of the resource prices times the resource requirements. What is the maximum number of units of the product you can produce, given a (positive) total budget B?

Express your answer using vector notation, in terms of the quantity and price n-vectors q and p, the budget B, and any basic arithmetic operations you might need. If you introduce additional vectors, be sure to say what they are, *i.e.*, what their entries are.

1.10 *Exponentially weighted average.* Suppose w is an n-vector with positive entries that sum to one, *i.e.*, $\mathbf{1}^T w = 1$. For any n-vector x, $w^T x$ can be interpreted as a weighted average of the entries of x. In this context the entries of w are called the weights. The weighted average with $w = (1/n)\mathbf{1}$ is the ordinary (or unweighted) average.

One common choice of weights, typically used when the vector x represents a time series, are *exponential weights*, defined by $w_i = (1/\alpha)\gamma^i$, where $\gamma > 0$ is a parameter and

$$\alpha = \gamma^1 + \cdots + \gamma^n$$

is the so-called normalizing constant, which ensures that $\mathbf{1}^T w = 1$. Using a well-known formula for the sum of powers, we have

$$\alpha = \begin{cases} \gamma\frac{1-\gamma^n}{1-\gamma} & \gamma \neq 1, \\ n & \gamma = 1. \end{cases}$$

With this choice of w, the weighted average $w^T x$ is called the *exponentially weighted average* of x, with parameter γ.

Here is an interpretation of the exponentially weighted average. For $\gamma < 1$, the associated weighted average emphasizes the entries of x with small indexes, and de-emphasized the entries of x with large indexes, when finding the average; for $\gamma > 1$ the opposite is true.

Here is a typical use of an exponentially weighted average. The registrar of a university needs to estimate the enrollment in the next offering of a certain class, using previous enrollment numbers. The vector x gives the previous enrollments in the class, with x_1 the most recent offering, x_2 the one before that, and so on. The estimate of the next offering enrollment is the weighted average of x with $\gamma < 1$, say, $\gamma = 0.8$. This means that the enrollment in the more recent offerings of the class have more weight in forming the average.

Finally, we get to the problem. Plot the exponential weights for $n = 50$ and the three parameter values $\gamma = 1$, $\gamma = 0.95$, $\gamma = 1.05$. Also plot the 50-vector x with $x_i = \sin(2\pi i/50)$.

Find the weighted average of the n-vector x for the three values of γ. Are the results consistent with the interpretation given above?

2 Linear functions

2.1 *Deviation of middle element value from average.* Suppose x is a n-vector, with $n = 2m-1$ and $m \geq 1$. We define the middle element value of x as x_m. Define

$$f(x) = x_m - \frac{1}{n}\sum_{i=1}^{n} x_i,$$

which is the difference between the middle element value and the average of the coefficients in x. Express f in the form $f(x) = a^T x$, where a is an n-vector.

2.2 *Nonlinear functions.* Show that the following two functions $f : \mathbf{R}^3 \to \mathbf{R}$ are *not* linear.

(a) $f(x) = (x_1 - x_2 + x_3)^2$.

(b) $f(x) = (x_1 + 2x_2 - x_3)_+$, where for any real number a, $(a)_+$ is the *positive part* of a, defined as $(a)_+ = \max\{a, 0\}$. *Remark.* This function is widely used in neural networks, where is it called the *relu* function, a shortening of *rectified linear unit.*

Hint. To show a function f is not linear, it is enough to find specific vectors x, y, and scalars α, β, for which superposition does not hold, *i.e.*,

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

It's polite to the reader, but not formally required, to find simple values x, y, α, β for your counterexample.

2.3 *Net present value.* Suppose that the n-vector c represents a cash flow over n periods. The NPV (net present value) of the cash flow, with (positive) per-period interest rate r, is defined as $\mathrm{NPV}(c, r) = c_1 + (1+r)^{-1}c_2 + \cdots + (1+r)^{-n+1}c_n$.

(a) How are $\mathrm{NPV}(c, 0.05)$ and $\mathrm{NPV}(2c, 0.05)$ related? (The second case is twice the cash flow, with the same 5% per-period interest rate.)

(b) How are $\mathrm{NPV}(c, 0.05)$ and $\mathrm{NPV}(c, 0.10)$ related? (The second case is the same cash flow, with twice the per-period interest rate.)

In both cases, your response can either be a specific and simple formula relating the two quantities, or the response 'It's complicated', which means that you cannot say what the relationship is, without knowing more about the entries of c.

2.4 Suppose we have a linear function $f : \mathbf{R}^3 \to \mathbf{R}$, and

$$a = (4, -2, 0), \quad b = (8, -4, 0), \quad c = (5, -1, 1).$$

You are told that $f(a) = 5$. What can you say about $f(b)$? If it cannot be determined, say so. If it can be determined from the information given above, give its value. Repeat for $f(c)$.

2.5 Let the T-vector x be a time series, with x_t its value in period t, for $t = 1, \ldots, T$. (This could be hourly rainfall, a daily stock price, or course enrollment each term, for example.) Consider the function $f(x) = x_T - \mathbf{avg}(x)$, the difference of the last value and the average. Is f linear? If f is not linear, give a specific example of T-vectors x, y, and constants α and β, for which $f(\alpha x + \beta y) \neq \alpha f(x) + \beta f(y)$. If f is linear, give a specific T-vector a for which $f(x) = a^T x$ holds for any T-vector x.

2.6 *Combining two regression models.* Two interns, Ann and Bob, construct regression models to predict the same quantity y, using different sets of features. Ann's model is

$$\hat{y}^{\text{ann}} = (x^{\text{ann}})^T \beta^{\text{ann}} + v^{\text{ann}},$$

where x^{ann} is an n^{ann}-vector of features, and Bob's model is

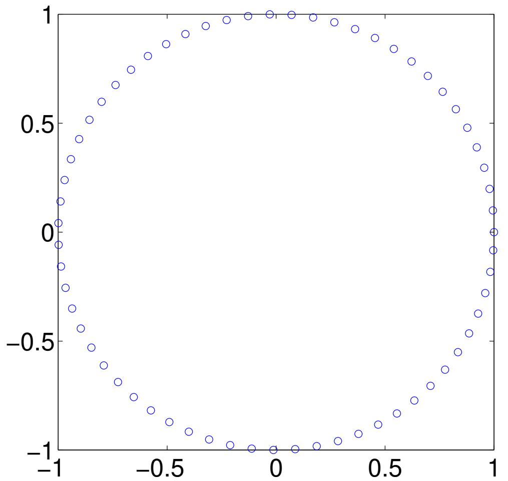
$$\hat{y}^{\text{bob}} = (x^{\text{bob}})^T \beta^{\text{bob}} + v^{\text{bob}},$$

where x^{bob} is an n^{bob}-vector of features. (The superscripts here are labels that allow us to distinguish the two different regression model coefficients and feature dimensions.) Their manager decides to combine their two estimates, weighting Ann's estimate 70% and Bob's estimate 30%, to get the combined prediction $\hat{y} = 0.7\hat{y}^{\text{ann}} + 0.3\hat{y}^{\text{bob}}$. (Presumably this means the manager trusts Ann's model more than Bob's.) Express this as a single regression model $\hat{y} = x^T\beta + v$, where $x = (x^{\text{ann}}, x^{\text{bob}})$. Give β and v in terms of β^{ann}, β^{bob}, v^{ann}, and v^{bob}, using standard vector notation.

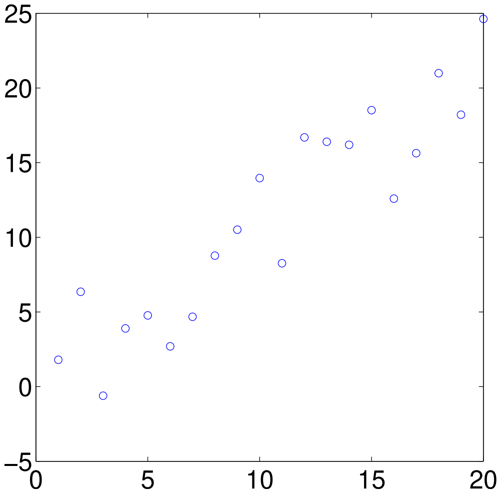
3 Norm and distance

3.1 *Correlation coefficient.* Each of the following plots shows the points corresponding to two vectors x and y of the same size, *i.e.*, we plot points at the locations (x_i, y_i). In each case, determine whether the correlation coefficient ρ of the two vectors is positive (and, say, ≥ 0.5), negative (say, ≤ -0.5), or near zero (say, less than 0.3 in absolute value). (You must choose one of these options.)

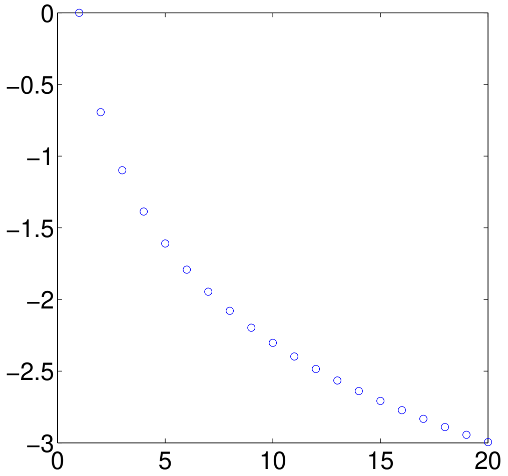
(a)



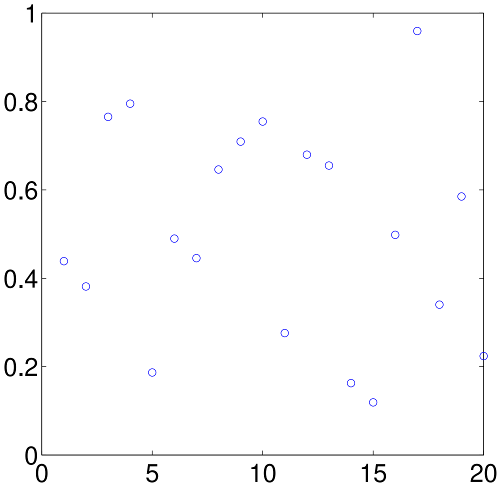
(b)



(c)



(d)



3.2 *Nearest neighbor and smallest angle.* Using Julia, find the nearest neighbor of $a = (1, 3, 4)$ among the vectors

$$x_1 = (4, 3, 5), \quad x_2 = (0.4, 10, 50), \quad x_3 = (1, 4, 10), \quad x_4 = (30, 40, 50).$$

Report the minimum distance of a to $x_1, \ldots, x_4$. Also, find which of $x_1, \ldots, x_4$ makes the smallest angle with a and report that angle.

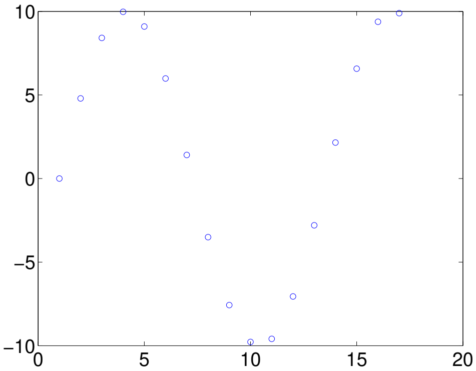
3.3 *Orthogonality.* Suppose the n-vectors a, b, and c satisfy $a \perp c$ and $b \perp c$. Which of the following statements must hold? (That is, are true for any a, b, and c that satisfy $a \perp c$, $b \perp c$.)

(a) $a \perp b$.

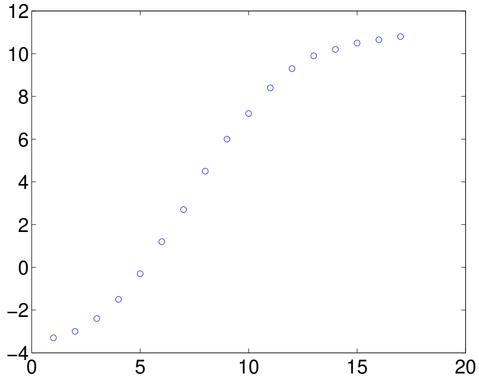
(b) $(a+b) \perp c$.

(c) $(a+c) \perp (b+c)$.

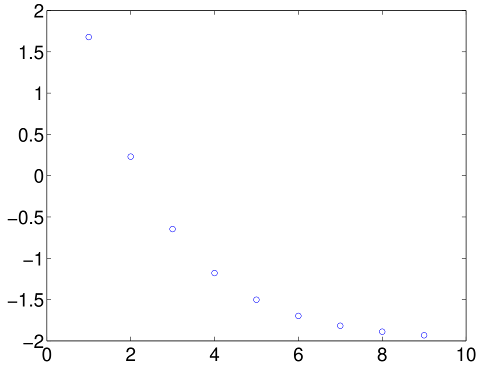
3.4 *Guessing means and standard deviations.* Each of the plots below shows a vector x, with x_i plotted on the vertical axis and i on the horizontal axis. For each case, estimate $\mathbf{avg}(x)$ and $\mathbf{std}(x)$. We are looking for a crude guess, say, within a factor of two.



(a)



(b)



(c)

3.5 Let α, β, and γ be scalars and let a, b, and c be pairwise orthogonal n-vectors. (This means that $a \perp b$, $a \perp c$, and $b \perp c$.) Express

$$\|\alpha a + \beta b + \gamma c\|$$

in terms of $\|a\|$, $\|b\|$, $\|c\|$, α, β, and γ.

3.6 *True or false.* Determine whether each of the following statements is true or false.

(a) If n-vectors x and y make an acute angle, then $\|x+y\| \geq \max\{\|x\|, \|y\|\}$.

(b) For any vector a, $\mathbf{avg}(a) \leq \mathbf{rms}(a)$.

3.7 Let $x = (1, 2, 3)$ and $y = (3, 2, 1)$. Find the quantities below. Your solutions can involve standard mathematical functions like squareroot, arc-cosine, and so on. You do not need to show any computations or give any justification.

(a) $\mathbf{rms}(x)$.

(b) $\mathbf{avg}(y)$.

(c) $\mathbf{std}(x)$.

(d) The correlation coefficient of x and y.

(e) The distance between x and y.

3.8 *Time to find the nearest neighbor.* We have a collection of 10000 different feature vectors, each of dimension 1000. A new feature vector (of dimension 1000) is given, and we need to determine which of the 10000 given feature vectors is its nearest neighbor. About how long would this take, on a computer capable of 10 Gflop/s? (That is, 10^{10} floating point operations per second.) Choose one of the responses below. Briefly justify your choice.

- Well under one second.
- A few seconds.
- A few minutes.
- Around an hour.

3.9 Suppose x is an n-vector, with $n > 1$. Then x^2

(a) is the inner product of x with itself

(b) is the vector with entries x_i^2

(c) makes no sense

(Choose one.)

3.10 Suppose x is a vector. Choose one in each subproblem below.

(a) $\|x\| = -\|-x\|$

(i) always

(ii) sometimes

(iii) never

(b) $\|(x, x)\| =$

(i) $\|x\|$

(ii) $2\|x\|$

(iii) $\sqrt{2}\|x\|$

(iv) $\frac{1}{2}\|x\|$

3.11 Can we generalize the triangle inequality to three vectors? That is, do we have

$$\|a+b+c\| \leq \|a\|+\|b\|+\|c\|$$

for any n-vectors a, b, c? If so, justify it. If it's not true, give a counter-example, *i.e.*, specific vectors a, b, and c for which the inequality is false.

3.12 *RMS value of difference of orthogonal vectors.* Suppose the n-vectors a and b are orthogonal. What is $\mathbf{rms}(a-b)$? If it cannot be determined without more information, say so. Otherwise give its value.

3.13 Suppose x is a 20-vector with $\|x\| = 10$. Which of the statements below follow from the Chebyshev inequality? (We do not count statements that are true, but don't follow from the Chebyshev inequality.)

(a) no x_i can satisfy $|x_i| \geq 10.5$

(b) at least one x_i must have magnitude at least 2

(c) no more than 3 entries of x can exceed 6 in magnitude

(d) no more than half the entries of x are positive

3.14 Which of the following are true? (True means it holds for any vector x.)

(a) $\mathbf{std}(x) \leq \mathbf{rms}(x)$

(b) $\mathbf{rms}(x) \leq |\mathbf{avg}(x)|$

(c) $\mathbf{std}(x)^2 = \mathbf{rms}(x)^2 - \mathbf{avg}(x)^2$

(d) $\mathbf{std}(x) = 0$ only when all entries of x are equal

3.15 Suppose x gives the daily temperature in Palo Alto and y gives the daily temperature in the neighboring city Menlo Park, over the same 5 year period. What would you expect the correlation coefficient between x and y to be? For example, would you expect it to be positive, negative, near 1, near -1, or near 0?

3.16 Suppose that $\|a+b\| < \|a\|$. What can you say about $\theta = \angle(a,b)$?

(a) $\theta = \pi/2$ (orthogonal)

(b) $\theta > \pi/2$ (obtuse angle)

(c) $\theta < \pi/2$ (acute angle)

Choose one and justify your choice. (The assumption implies that both a and b are nonzero. You can use this fact without justification.)

3.17 Suppose x and y are Boolean feature vectors, *i.e.*, each entry is either 0 or 1, that encode the presence of symptoms in patients Alice and Bob. Which of the following are true statements?

(a) $x^T y$ is number of symptoms Alice and Bob have in common

(b) $\|x\|^2$ is number of symptoms Alice has

(c) $\mathbf{1}^T y$ is number of symptoms Bob has

(d) $\|x-y\|^2$ is number of symptoms Alice has but Bob does not

(e) $\mathbf{1}^T(x-y)$ is number of symptoms Alice has but Bob does not

(f) $x \perp y$ means that Alice and Bob do not share any symptoms

3.18 Stocks with ticker symbols AAA and BBB have daily returns over three days given by $a = (+0.01, -0.01, +0.03)$ and $b = (-0.01, 0.00, +0.02)$. Which of the following must be true?

(a) AAA has higher (mean) return than BBB (over these three days)

(b) BBB has rower risk (standard deviation of return) than AAA (over these three days)

(c) AAA is a better investment than BBB

(d) AAA and BBB are likely in the same industry sector

3.19 Let x and y be (nonzero) word count vectors for two documents, associated with a given dictionary. Choose one in each of the subproblems below, and briefly justify your answer.

(a) If $\angle(x,y) = 0$, we can conclude

(i) the documents are identical

(ii) the documents share no dictionary words

(iii) the word count histogram vectors of the two documents are the same

(iv) the documents have the same author

(v) the documents have different authors

(b) If $\angle(x,y) = 90^\circ$, we can conclude

(i) the documents are identical

(ii) the documents share no dictionary words

(iii) the word count histogram vectors of the two documents are the same

(iv) the documents have the same author

(v) the documents have different authors

3.20 *Nearest neighbors of Wikipedia word count histograms.* In this problem you will explore the nearest neighbors of the word count histograms of a collection of 500 Wikipedia articles. To get the data, run the Julia commands

```
using VMLS
articles, dictionary, titles = wikipedia_data();
```

- `articles` contains the word histogram vectors of 500 articles; each word histogram vector has 4423 elements.
- `dictionary` is an array of 4423 words.
- `titles` is an array of the 500 article titles.

To access entries of these vectors, use `[i]`. For example, `articles[3]` is the word histogram vector for the 3rd article, `dictionary[15]` is the 15th word in the dictionary, and `titles[6]` is the title of the 6th article. (We encourage you to try these.)

To find the k nearest neighbors among a give list of vectors `X` of the vector `u`, you can use the function `k_nearest_neighbors(X,u,k)` (included in the VMLS package). You are encouraged to look at the code for this function, which is very short and straightforward.

Here are some useful Julia tips.

- The function `sortperm` gives the indices when you sort a vector. `sortperm(articles[i],rev=true)` gives the indices of words when they are sorted from largest to smallest histogram value in article i. `sortperm(articles[i],rev=true)[1:10]` gives the indices of the 10 most common words in article i.
- `dictionary[sortperm(articles[i],rev=true)[1:10]]` gives the words in the dictionary, sorted from largest to smallest histogram value, in article i.
- The function `k_nearest_neighbors` (in the VMLS package) can be used to find the k nearest neighbors of a vector from among a list of vectors:

  ```
  indices, distances = k_nearest_neighbors(X, u, k)
  ```

  Here `X` is a list of vectors, `u` is a vector, and `k` is the number of nearest neightbors. `indices` gives the indices of the k nearest neightbors, sorted from closest to farthest; `distances` gives the corresponding distances.

(a) *Choose an article.* Let i be your favorite integer between 1 and 500. Find the title of the article, and the 10 most common words in it.

(b) *Find its 10 nearest neighbors.* Find the titles of the 10 nearest neighbors of your article. (More precisely: find the titles of the 10 articles whose word count histogram vectors are closest to the word count histogram of your chosen article.) Do the results makes sense?

Hint. If the results don't make sense, then choose another favorite integer, and never speak of this incident again.

3.21 *Checking some VMLS functions.* The VMLS package contains the functions `rms`, `avg`, `std`. Check these by evaluating them on $x = (1, 0, -2, 3)$, and comparing the result to a direct calculation that does not use the VMLS functions. For example, you might compare `rms(x)` to `sqrt(sum(x.^2)/4)`, or `norm(x)/sqrt(4)`. Your direct calculations can involve a few lines of code.

3.22 *Checking some inequalities.* Generate two random 10-vectors a and b (using `randn(10)`). Check the following inequalities by evaluating the lefthand and righthand sides and verifying that the inequality holds for your a and b.

(a) The triangle inequality, $\|a + b\| \leq \|a\| + \|b\|$.

(b) The Cauchy-Schwarz inequality, $|a^T b| \leq \|a\|\|b\|$.

3.23 *Checking the Chebyshev inequality.* Generate a random 200-vector x using `x=randn(200)`. Verify that no more than 8 of the entries of x satisfy $|x_i| \geq 5\,\mathbf{rms}(x)$.

3.24 *Triangle equality.* Suppose the two nonzero n-vectors a and b satisfy $\|a + b\| = \|a\| + \|b\|$, *i.e.*, the triangle inequality holds with equality. Geometric intuition suggest that this only happens when the vectors are aligned, *i.e.*, satisfy $\angle(a, b) = 0$ (which means each is a positive multiple of the other). Show that this is the case. *Hint.* Square both sides of the equality.

3.25 *Stock return time series.* The 250-vector r is the return times series of a stock, denoted in percent, *i.e.*, $r_{37} = -0.8$ means that the stock price went down 0.8% on day 37, and $r_{230} = 1.2$ means that the stock price went up 1.2% on day 230. (There are around 250 trading days in one year, so r gives the daily returns over one year.) The standard deviation of r (*i.e.*, its risk) is σ, and its mean (*i.e.*, return) is μ. (These are called the daily risk and return; most people work with so-called annualized risk and return.)

(a) What is the RMS value of r?

(b) Use the Chebyshev inequality to give a maximum number of days for which we can have $|r_i| \geq 3$, *i.e.*, a gain or loss of 3% or more.

(c) Find the two quantities in parts (a) and (b) for $\mu = 0.08$ and $\sigma = 1.2$. (These are realistic values.) Please give your answers to four decimal places.

3.26 *Finding a similar bond to buy.* A bond (which is an asset that you can buy or sell) is associated with an n-vector b, whose entries give important features of the bond related to its coupon payment schedule, its sector and region, whether its payments are tax exempt, and its ratings by various agencies. (You do not need to know what any of these things mean.) Each of these features has been shifted and scaled so their typical values are all in the same range, roughly between -1 and 1.

A bond trader wishes to buy a bond with associated feature vector b^{des}, but the particular bond she wants is not available. She has a list of N bonds that *are* available to purchase, with feature vectors $b_1, \ldots, b_N$. (This is a list of N n-vectors.)

From among these bonds that are available, which might you suggest that she buy? Your suggestion should be in the language of VMLS, and can use words like inner product, norm, linear combination, angle, distance, correlation, nearest neighbor, k-means, and so on.

3.27 *A pth moment deviation bound.* The Chebyshev inequality states that the number k of elements of an n-vector x satisfying $|x_i| \geq a$ must satisfy $\frac{k}{n} \leq \frac{\|x\|^2}{a^2 n}$. We can define *p-norms*, which are distinct from Euclidean norms, for $1 \leq p < \infty$ by

$$\|x\|_p = \sqrt[p]{|x_1|^p + |x_2|^p + \cdots + |x_n|^p},$$

so $\|x\|_p^p = |x_1|^p + \cdots + |x_n|^p$. In analogy with **rms**, let $\overline{x}_n = \frac{1}{n}(x_1 + \cdots + x_n) = \frac{1}{n}\mathbf{1}^T x$ denote the mean of x, and define the "root-mean-pth" value

$$\textbf{rmp}(x) = \sqrt[p]{\frac{1}{n}\left(|x_1 - \overline{x}_n|^p + \cdots + |x_n - \overline{x}_n|^p\right)} = \frac{1}{n^{1/p}}\left\|x - (\mathbf{1}^T x/n)\mathbf{1}\right\|_p.$$

(a) Using a similar argument to that for the Chebyshev inequality, verify the following *pth moment* bound: if k is the number of elements of the n-vector x satisfying $|x_i| \geq a$, then

$$\frac{k}{n} \leq \frac{\|x\|_p^p}{na^p}.$$

(b) Verify the following *mean pth moment bound*: if k is the number of elements of the n-vector x satisfying $|x_i - \frac{1}{n}\mathbf{1}^T x| \geq a$, then

$$\frac{k}{n} \leq \frac{\textbf{rmp}(x)^p}{a^p}.$$

(c) According to the mean pth moment bound, what is the largest percentage of entries in a vector x that can be farther than $2\mathbf{rmp}(x)$ units away from $\mathbf{avg}(x)$ for $p = 2$?

(d) According to the mean pth moment bound, what is the largest percentage of entries in a vector x that can be farther than $2\mathbf{rmp}(x)$ units away from $\mathbf{avg}(x)$ for $p = 4$?

(e) According to the mean pth moment bound, what is the largest percentage of entries in a vector x that can be farther than $2\mathbf{rmp}(x)$ units away from $\mathbf{avg}(x)$ for $p = 8$?

4 Clustering

4.1 *Building a recommendation engine using k-means.* A set of N users of a music-streaming app listens to songs from a library of n songs over some period (say, a month). We describe user i's listening habits by her playlist vector, which is the n-vector p_i defined as

$$(p_i)_j = \begin{cases} 1 & \text{user } i \text{ has played song } j \\ 0 & \text{user } i \text{ has not played song } j, \end{cases}$$

for $j = 1, \ldots, n$. (Note that p_i is an n-vector, while $(p_i)_j$ is a number.) You can assume that if a user listens to a song, she likes it.

Your job (say, during a summer internship) is to design an algorithm that recommends to each user 10 songs that she has not listened to, but might like. (You can assume that for each user, there are at least 10 songs that she has not listened to.)

To do this, you start by running k-means on the set of playlist vectors $p_1, \ldots, p_N$. (It's not relevant here, but a reasonable choice of k might be 100 or so.) This gives the centroids $z_1, \ldots, z_k$, which are n-vectors.

Now what do you do? You can explain in words; you do not need to give a formula to explain how you make the recommendations for each user.

4.2 *Topic discovery via k-means.* In this problem you will use k-means to cluster 500 Wikipedia articles selected from 5 broad groups of topics. To get the data, run the Julia commands

```
using VMLS
articles, dictionary, titles = wikipedia_data();
```

- `articles` contains the word histogram vectors of 500 articles; each word histogram vector has 4423 elements.
- `dictionary` is an array of 4423 words.
- `titles` is an array of the 500 article titles.

To access entries of these vectors, use `[i]`. For example, `articles[3]` is the word histogram vector for the 3rd article, `dictionary[15]` is the 15th word in the dictionary, and `titles[6]` is the title of the 6th article. (We encourage you to try these.)

You can run the k-means algorithm on the group of vectors in `articles` using the `kmeans()` function (from the VMLS package). For example,

```
assignment, reps = kmeans(articles, 3)
```

will run k-means on the vectors in `articles` with $k = 3$.

- The array `assignment` gives the cluster group assignments for the articles. For example, `assignment[32]` tells us which cluster article 32 belongs to (which is an integer).
- `reps` gives the group representatives. For example, `reps[1]` is the group representative for group 1 (which is a 4423-vector).

(a) Run k-means on the data with $k = 5$.

(b) Construct a 500-vector `d` that contains the distance between each vector and its representative.

(c) Consider cluster number 1. Give the 5 most common words that appear in articles in this cluster. *Hints.*

- The function `sortperm` gives the indices when you sort a vector. `sortperm(reps[3],rev=true)` gives the indices of words when they are sorted from largest to smallest histogram value in representive 3.
- `dictionary[sortperm(reps[3],rev=true)]` gives the words in the dictionary, sorted from largest to smallest in representative 3.

(d) Find the 5 articles that are closest to representative 1. Feel free to look up these articles in Wikipedia.

(e) Repeat parts (c) and (d) for the other clusters.

4.3 *Centroid interpretations.* The n-vectors $x_1, \ldots, x_N$ contain n attributes of N patients admitted to a hospital. The first component, $(x_i)_1$, is the age of patient i. The second component, $(x_i)_2$, is 1 if the patient is having trouble breathing, and 0 if not. (The other components give other attributes.) An ENGR108 graduate carries out k-means on this data set, with $k = 22$. She finds the 5th centroid or group representative is $z_5 = (41.6, 0.37, 1.55, \ldots, 29.6)$.

Give a simple short interpretation in English of the first two components, *i.e.*, $(z_5)_1 = 41.6$ and $(z_5)_2 = 0.37$.

4.4 Which of the following statements are correct?

(a) The goal of clustering a set of vectors is to choose the best vectors from the set

(b) The goal of clustering a set of vectors is to divide them into groups of vectors that are near each other

(c) The goal of clustering a set of vectors is to determine the nearest neighbors of each of the vectors

(d) The k-means algorithm always converges to a clustering that minimizes the mean-square vector-representative distance

(e) The k-means algorithm can converge to different final clusterings, depending on the initial choice of representatives

(f) The k-means algorithm is widely used in practice

(g) The choice of k, the number of clusters to partition a set of vectors into, depends on why you are clustering the vectors

(h) The choice of k, the number of clusters to partition a set of vectors into, should always be as large as your computer system can handle

4.5 *Clustering class grade data.* In a class with 300 students the homework assigments, quizzes, and exams all together include 200 different problems or questions, each graded on a scale of 0–10. The teaching assistants (TAs) cluster this data using k-means in two different ways described below.

(a) *Clustering the problems.* The TAs form 200 300-vectors $q_1, \ldots, q_{200}$, each one associated with one of the 200 questions, with $(q_i)_j$ the grade of student j on question i. The TAs then cluster this collection of vectors into $k = 5$ groups using k-means.

(b) *Clustering the students.* The TAs form 300 200-vectors $s_1, \ldots, s_{300}$, each one associated with one of the 300 students, with $(s_i)_j$ the grade of student i on question j. The TAs then cluster this collection of vectors into $k = 10$ groups using k-means.

For each of these, give a short description in English of what the TAs might find. For example, describe some clusters they might find. Your responses are of course pure conjecture, but they should sound plausible.

5 Linear independence

5.1 *Linear independence under combination.* Suppose $S = \{a, b, c\}$ and $T = \{d, e, f\}$ are two linearly independent sets of n-vectors. For each of the sets given below, determine which statement is correct. You may not use a computer to answer the questions. (Only one is correct in each case.)

(a) $\{a, b, c, d, e, f\}$

- is always linearly independent.
- is always linearly dependent.
- could be linearly independent or linearly dependent, depending on the values of $a, \ldots, f$.

(b) $\{a + d, b + e, c + f\}$

- is always linearly independent.
- is always linearly dependent.
- could be linearly independent or linearly dependent, depending on the values of $a, \ldots, f$.

(c) $\{a, a + b, a + b + c\}$

- is always linearly independent.
- is always linearly dependent.
- could be linearly independent or linearly dependent, depending on the values of $a, \ldots, f$.

5.2 *Order of vectors in the Gram-Schmidt algorithm.* Suppose a_1, a_2 is a list of two linearly independent n-vectors. When we run the Gram-Schmidt algorithm on this list, we obtain the orthonormal vectors q_1, q_2.

Now suppose we run the Gram-Schmidt algorithm on the list of vectors a_2, a_1 (*i.e.*, the same vectors, in reverse order). Do we get the orthonormal vectors q_2, q_1 (*i.e.*, the orthonormal vectors obtained from the original list, in reverse order)?

If you believe this is true, give a very brief explanation why. If you believe it is not true, give a simple counter-example.

5.3 Suppose q_1, q_2, and q_3 are orthonormal n-vectors.

(a) What can you conclude about n?

(b) For what values of n does $x = (q_1^T x)q_1 + (q_2^T x)q_2 + (q_3^T x)q_3$ hold, for all n-vectors x?

5.4 *Checking linear independence using the Gram-Schmidt algorithm.* The VMLS package includes a simple implementation of the Gram-Schmidt algorithm. (You are free and even encouraged to look at the definition of this function.) This function accepts a list of vectors and returns a list of the orthonormal vectors. It gives a warning if Gram-Schmidt terminates early, which means the list of vectors in linearly dependent.

In Julia, a list of vectors can be specified using, *e.g.*,

```
list_of_vectors = ( [1,2], [3,4], [5,6] )
```

Are the following sets of vectors linearly dependent or linearly independent? If you can say ahead of time that the list is linearly dependent due to the dimension-independence inequality, say so. (But still check it using `gram_schmidt`.)

(a) $(1,-1.1),\ (-2.8,-0.3),\ (-0.4,1.5)$.

(b) $(1,0,1,0,1),\ (0,1,0,0,1),\ (0,1,0,1,0)$.

(c) $(1,2,0),\ (-2,0,3),\ (1,0,2)$.

(d) $(1,2,0),\ (-2,0,3),\ (-1,2,3)$.

5.5 *Fourier basis and expansion.* A very famous orthonormal basis for n-vectors is the *Fourier basis*, named after the mathematician Joseph Fourier. For n even, the basis has the form

$$d,\ s_1,\ c_1,\ s_2,\ c_2,\ \ldots,\ s_{n/2},\ c_{n/2},\ a.$$

Here $d = \mathbf{1}/\sqrt{n}$ is the *constant basis vector*, a is the *alternating basis vector*, given by

$$a_j = \left(\frac{1}{n}\right)^{1/2}(-1)^j, \quad j = 1,\ldots,n,$$

s_i is the ith *sine basis vector*, given by

$$(s_i)_j = \left(\frac{2}{n}\right)^{1/2}\sin\left(\frac{2\pi i j}{n}\right), \quad i = 1,\ldots,n/2, \quad j = 1,\ldots,n,$$

and c_i is the ith *cosine basis vector*, given by

$$(c_i)_j = \left(\frac{2}{n}\right)^{1/2}\cos\left(\frac{2\pi i j}{n}\right), \quad i = 1,\ldots,n/2, \quad j = 1,\ldots,n/2-1.$$

(Note that the constant basis vector can be thought of as the cosine basis vector with $i = 0$, scaled by $\sqrt{2}$; the alternating basis vector can be thought of as the cosine basis vector with $i = n/2$, scaled by $\sqrt{2}$.)

When an n-vector x is expressed as a linear combination of the Fourier basis vectors, it is referred to as the *Fourier expansion* or *spectral expansion* of x. The coefficients are called the *Fourier coefficients*. This comes up in many different applications.

(a) We are not asking you to slog through the algebra to show that the Fourier basis is orthonormal. Instead, please check it numerically, by evaluating the norm of a few of them (they should be one, or very near one due to numerical errors), and checking the inner product of a few pairs of different basis elements (these should be zero, or very small due to numerical errors).

(b) Take $n = 50$. Plot d, s_1, c_1, s_2, c_2, and a.

(c) Give the Fourier expansion of the 6-vector x, with $x_i = i$, $i = 1,\ldots,6$. That is, find its Fourier coefficients, and verify that the associated linear combination of the Fourier basis vectors is x (within a small numerical error). (You can do this numerically.)

Hint. You might find the following code useful.

```
function fourier_basis_const(n)
return(ones(n)/sqrt(n))
end
```

```
function fourier_basis_sin(i,n)
return((sqrt(2/n))*sin.((2*i*pi/n)*(1:n)))
end
function fourier_basis_cos(i,n)
return((sqrt(2/n))*cos.((2*i*pi/n)*(1:n)))
end
function fourier_basis_alt(n)
return((sqrt(1/n))*(-1).^(1:n))
end
```

5.6 Suppose $q_1, \dots, q_k$ is an orthonormal set of n-vectors, with $k < n$. We wish to determine if a given n-vector a is a linear combination of them.

An intern (an ENGR108 alumna) claims that this is easy to do. She says that we compare the n-vector

$$\hat{a} = (q_1^T a)q_1 + \cdots + (q_k^T a)q_k$$

to a. If $\hat{a} = a$, then a is a linear combination of $q_1, \dots, q_k$. If $\hat{a} \neq a$, then a is a not linear combination of $q_1, \dots, q_k$.

Is the intern right? Give your answer as yes or no, and add a very short explanation.

5.7 *Projections.* The *projection* $\mathbf{proj}_S(y)$ of an n-vector y onto a set S of n-vectors is the point $s \in \mathbf{R}^n$ in S closest to y as measured by the distance $\|y - s\|$, that is, $\mathbf{proj}_S(y)$ is the point in S minimizing $\|y - s\|^2$ over $s \in S$.

(a) Let q be a vector with $\|q\| = 1$. Find the $t \in \mathbf{R}$ minimizing

$$\|y - tq\|^2 .$$

It may be useful to recall the value t minimizing the quadratic $f(t) = at^2 + bt + c$ when $a > 0$.

(b) Let $S = \{tq \mid t \in \mathbf{R}\}$ be the linear span of q. Give $\mathbf{proj}_S(y)$.

(c) Draw a picture of your result in part (b). Your picture should include the vector $\mathbf{proj}_S(y)$, y, and $y - \mathbf{proj}_S(y)$. What is $(y - \mathbf{proj}_S(y))^T q$? *Hint.* The answer to the last part is why we call these *orthogonal projectors.*

5.8 *Projections onto the span of orthonormal vectors.*

(a) Let the n-vectors $\{q_1, \dots, q_k\}$ be orthonormal and $t_1, \dots, t_k$ be scalars. Justify the equality

$$\|y - t_1 q_1 - \cdots - t_k q_k\|^2 = \|y\|^2 - 2(t_1 q_1^T y + \cdots + t_k q_k^T y) + t_1^2 + \cdots + t_k^2.$$

(b) Let $S = \{t_1 q_1 + \cdots + t_k q_k \mid t_i \in \mathbf{R}, i = 1, \dots, k\}$ be the linear span of $\{q_1, \dots, q_k\}$. Give $\mathbf{proj}_S(y)$. *Hint.* Find the values $t_1, \dots, t_k \in \mathbf{R}$ minimizing $\|y - t_1 q_1 - \cdots - t_k q_k\|^2$.

(c) In a sentence or so, what does this say about the iterations of Gram-Schmidt?

5.9 *Orthogonal projections.* Let $u \in \mathbf{R}^n$ be a unit vector, so that $\|u\| = 1$, and $x \in \mathbf{R}^n$ be any other vector. Let $L = \{tu \mid t \in \mathbf{R}\}$ be the line through the origin in the direction u.

(a) Suppose that v is a vector on the line L, that is, $v \in L = \{tu \mid t \in \mathbf{R}\}$, and that we have the orthogonality condition

$$(x - v)^T u = 0.$$

Give a formula for v in terms of x and u.

(b) Again, assume that $v \in L$ satisfies $(x - v)^T u = 0$. Show that if $t \neq 0$, then

$$\|x - v - tu\|^2 > \|x - v\|^2 .$$

(c) Draw a picture illustrating the following claim, which you have proved: the point $v \in L$ is the closest point to x on the line L if and only if the direction from x to v is orthogonal to u, that is, $(x - v)^T u = 0$.

6 Matrices

6.1 *Checking superposition in Julia.* Generate a random 20×10 matrix A, as well as 10-vectors x and y, and scalars α and β. Evaluate the two 20-vectors $A(\alpha x + \beta y)$ and $\alpha(Ax) + \beta(Ay)$, and verify that they are very close by printing the norm of the difference. (If the numerical calculations were done exactly, they would be equal. Due to very small rounding errors made in the floating-point calculations, they will not be exactly equal.)

Hint. The Julia function `rand` can be used to generate random scalars, vectors, and matrices. `rand()` generates a random number, `rand(n)` generates a random n-vector, and `rand(m,n)` generates a random $m \times n$ matrix.

6.2 *Vandermonde matrices in Julia.* Write a function that takes a positive integer n and an m-vector t as inputs and generates the corresponding $m \times n$ Vandermonde matrix.

6.3 *Linear independence.* For each of the following matrices, determine which response is correct. You may not use a computer to answer the questions.

(a)

$$\begin{bmatrix} 428 & 973 & -163 & 245 & -784 & 557 \\ 352 & 869 & 0 & 781 & -128 & 120 \\ 1047 & 45 & -471 & 349 & -721 & 781 \end{bmatrix}$$

- The columns are linearly independent.
- The columns are linearly dependent.
- This is not an appropriate question.

(b)

$$\begin{bmatrix} 768 & 1121 & 3425 & 8023 \\ -2095 & -9284 & 5821 & -6342 \\ 4093 & -3490 & -7249 & 8241 \\ 834 & 1428 & 4392 & 5835 \\ -7383 & 1435 & 2345 & -293 \end{bmatrix}$$

- The columns are linearly independent.
- The columns are linearly dependent.
- This is not an appropriate question.

6.4 *Difference from trailing three-day rolling average.* The n-vector p gives the daily time series of the price of an asset over n trading days, with $n \geq 4$. The $(n-3)$-vector d gives the difference of the current asset price and the average asset price over the previous three trading days, starting from the fourth day. Specifically, for $i = 1, \ldots, n-3$, we have $d_i = p_{i+3} - (p_i + p_{i+1} + p_{i+2})/3$. (Note that d is an $(n-3)$-vector.)

Give the matrix A for which $d = Ap$, for the specific case $n = 6$. Be sure to give its size and all entries.

Remark. Time series similar to d are used in some simple trading schemes, which buy or sell the asset depending on whether the price is high or low compared to a trailing multi-day rolling average.

6.5 Let $s = A^T\mathbf{1}$. What is s_i, the ith element of s? Express your answer in English.

6.6 Let P be an $m \times n$ matrix and q be an n-vector, where P_{ij} is the price of good j in country i, and q_j is the quantity of good j needed to produce some product. Describe $(Pq)_i$ in English.

6.7 Suppose A is an $n \times n$ matrix. When is $A + A^T$ symmetric? You can answer never, sometimes, or always. Justify your anwwer.

6.8 Let A be a matrix, and let $B = \begin{bmatrix} A \\ I \end{bmatrix}$. What can you say about the linear independence of the rows of B? What can you say about the linear independence of the columns of B? In either case if the answer depends on the details of the matrix A, explain.

6.9 Suppose $Ax = 0$, where A is an $m \times n$ matrix and x is a nonzero n-vector. Then we can conclude (choose one)

(a) $A = 0$.

(b) the rows of A are linearly dependent.

(c) the columns of A are linearly dependent.

('We can conclude X' means that statement X is always true, with no further assumptions about A.)

6.10 Suppose $y = (x, x, x)$, where x is an n-vector. Give the matrix A for which $y = Ax$ always holds. Be sure to say that the dimensions of A are.

6.11 *Constructing matrices in Julia.* In each part below, use Julia to create the matrix A, described by its effect on an arbitrary vector x as $y = Ax$. Check that the matrix you create has the correct behavior when it multiplies a randomly generated vector x.

(a) x is a 5-vector; $y = x_{1:3}$.

(b) x and y are 4-vectors; y has the entries of x in reverse order.

(c) x is a 4-vector, and y is the 3-vector of differences of consecutive entries of x:

$$y = (x_2 - x_1, x_3 - x_2, x_4 - x_3)) = x_{2:4} - x_{1:3}.$$

(d) x and y are 4-vectors, and y_i is the difference between x_i and the average of the other 3 entries of x. For example, $y_2 = x_2 - (x_1 + x_3 + x_4)/3$.

6.12 *Running average matrix.* Suppose the T-vector x represents a time series, with x_t its value in period t, for $t = 1, \ldots, T$. The *running average* of x is another vector y, with

$$y_t = \frac{1}{t}\left(x_1 + \cdots + x_t\right), \quad t = 1, \ldots, T.$$

(For example, $y_1 = x_1$ and $y_T = \mathbf{avg}(x)$.)

(a) Find the $T \times T$ matrix R for which $y = Rx$. (We can call R the *running sum matrix.*) (To describe R you can say what R_{ij} is, for $i, j = 1, \ldots, T$.)

(b) Which of the following attributes does R have? Symmetric, diagonal, lower triangular, upper triangular. (You can choose more than one.)

(c) Give a brief interpretation of your answer to part (b) that does not refer to matrices, but only to properties of forming the running average.

6.13 *Vec and reshape.* The Julia command `vec(A)` takes the entries of an $m \times n$ matrix A, and puts them into an mn vector. It uses so-called *column-major order*, which means it stacks the first column of A on top of the second, on top of the third, and so on.

The Julia command `reshape` takes the entries of a vector or matrix, and puts them into a new matrix with different dimensions. It has the form `B = reshape(A,(p,q))`, where A is an $m \times n$ matrix, B is a $p \times q$ matrix, and $mn = pq$. This also uses column-major order, *i.e.*, it takes the columns one by one from A, and assembles the matrix B column by column. When a is a vector (*i.e.*, an $m \times 1$ matrix), `reshape(a,(p,q))` (with $pq = m$) re-arranges the entries of a into a matrix. This is the opposite of the `vec` command.

(a) Apply `vec` to

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

and call the result a. Reshape a into a 2×3 matrix, and verify that you get A.

(b) In a single line of code, express the matrix

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \end{bmatrix}$$

using only range construction (*i.e.*, `1:10`), `reshape`, and transpose.

6.14 *Householder reflections.* Let $u \in \mathbf{R}^n$ be a unit vector, meaning that $\|u\| = 1$. The *Householder reflection* of a vector $x \in \mathbf{R}^n$ over the direction u is

$$H_u(x) = 2(u^T x)u - x.$$

(a) Is H_u a linear function of x?

(b) Write $H_u(x) = Ax$ for a matrix A. Specify the dimensions of A, and give a formula for its entries A_{ij}.

(c) How many flops does computing $H_u(x)$ require? Is it fewer than multiplying Bx for an arbitrary $n \times n$ matrix B?

(d) Show that

$$H_u(x) = x + 2(uu^T x - x).$$

(e) Draw a picture of the Householder reflection in two dimensions. Your picture should include a vector x, the vector u, and the result $y = H_u(x)$. Why do we call this a reflection? *Hint.* It may be useful to recall Exercise 5.7.

6.15 *Householder reflections for annihilation.* Let us make a particular choice for the vector u in Exercise 6.14 above. Let x be a unit vector, that is, satisfying $\|x\| = 1$. Let $e_1 = (1, 0, \ldots, 0)$ be the first standard basis vector.

(a) Show that you can choose a sign $s \in \{-1, 1\}$ so that $\|x + se_1\|^2 \geq 2$.

For the s from part (a), define

$$u = \frac{x + se_1}{\|x + se_1\|}.$$

(b) Show that $H_u(x) = se_1$ whenever $\|x + se_1\| > 0$.

(c) Draw a picture of your result, which should include the vectors e_1, u, x, and $y = H_u(x)$.

(d) What have we annihilated?

6.16 Let A be an $m \times n$ matrix whose (i, j) entry is a_{ij}. Give two vectors $u \in \mathbf{R}^m$ and $v \in \mathbf{R}^n$ such that $u^T Av = \sum_{i=1}^m \sum_{j=1}^n a_{ij}$.

7 Matrix examples

7.1 *Equalization in communication.* Run the file `channel_equalization_data.jl`, which will define a message s, a channel c, and an equalizer h. (Your are welcome to look inside the file to see how we designed the equalizer.)

Plot c, h, and $h * c$. Make a brief comment about the channel and equalized channel impulse responses.

Plot s, y, and $\tilde{y}$ over the index range $i = 1, \ldots, 100$. Is it clear from this plot that $\hat{s} = \mathbf{round}(y_{1:N})$ will be worse estimate of s than $\hat{s}^{\text{eq}} = \mathbf{round}(\tilde{y}_{1:N})$?

Report the BER for $\hat{s}$ (estimating the message without equalization), and for $\hat{s}^{\text{eq}}$ (estimating the message with equalization).

Hint: To round a real vector `x` to $\{0, 1\}$ in Julia you can use `(x .> 0.5)`, which yields a Boolean vector. You can convert it to an integer vector (say, for plotting) by multiplying by 1. That is, `1*(x .> 0.5)`.

7.2 *Convolution in Julia.* Use Julia's `conv()` function to find the coefficients of the polynomial $(1 - x + 2x^2)^4$. *Hint.* Convolution gives the coefficients of the product of two polynomials.

7.3 *Audio filtering.* When the vector x represents an audio signal, and h is another (usually much shorter) vector, the convolution $y = h * x$ is called the *filtered* version of x, and h is called the *filter impulse response.* Filters can be used to smooth out audio signals (which reduces high frquency sounds and enhances low frequency sounds), or to sharpen them (which enhances high frequency sounds and reduces low frequency sounds), as in audio bass and treble tone controls. In this problem you will experiment with, and listen to, the effects of several audio filters.

The file `audio_filtering_original.wav` contains a 10-second recording with sample rate of $f = 44100/\text{sec}$. We let x denote the 441000-vector representing this recording. You can read in x and the sample rate f using the following code:

```
Pkg.add("WAV")
using WAV
x, f = wavread("audio_filtering_original.wav");
x = vec(x);
```

To play the signal, run:

```
wavplay(x, f);
```

If this not supported on your system, you can write the signal into a file, download the file from JuliaBox if you are using that, and then listen to it on your machine:

```
wavwrite(x, f, "filename.wav");
```

(a) *1ms smoothing filter.* Let h^{smooth} be the 44-vector $h^{\text{smooth}} = \frac{1}{44}\mathbf{1}_{44}$. (The subscript 44 gives the length of the vector.) The signal $h^{\text{smooth}} * x$ is the 1ms moving average of the input x. We can construct the vector h^{smooth} and compute the output signal as follows:

```
h_smooth = 1 / 44 * ones(44);
output = conv(h_smooth, x);
wavplay(output, f);
```

Listen to the output signal and briefly describe the effect of convolving h^{smooth} with x in one sentence.

(b) *Echo filter.* What filter (*i.e.*, vector) h^{echo} has the property that $h^{\text{echo}} * x$ consists of the original recording, plus an echo of the original recording 0.25 seconds delayed, with half the original amplitude? Since sound travels at about 340m/s, this is equivalent to the effect of hearing an echo from a wall about 42.5m away. Construct h^{echo} using Julia and listen to the output signal $h^{\text{echo}} * x$ to confirm the effect. Form and listen to the signal $h^{\text{echo}} * h^{\text{echo}} * x$ and very briefly describe what you hear.

Hint. The entries of the output signal $y = h^{\text{echo}} * x$ satisfy $y_i = x_i + 0.5x_{i-k}$, where we take $x_j = 0$ for j outside the range $1, \ldots, 441000$, and k is the number of samples in 0.25 seconds.

7.4 *Another property of convolution.* Suppose that a is an n-vector and b is an m-vector that satisfy $a * b = 0$. Is it true that either $a = 0$ or $b = 0$? If yes, explain why. If no, give a specific example of a and b, not both zero, with $a * b = 0$.

7.5 *Convolution.* What is $(1, -1) * (1, 0, 1)$?

7.6 *Mapping from commitments to cash flow.* An investor makes commitments to deals in each year $t = 1, \ldots, T$, in the amounts $u_1, \ldots, u_T$, in USD (or more realistically, millions of USD). (These are nonnegative, but that won't matter to us.) Commiting an amount u_t in year t entitles the investor to a cash flow of the form

$$(0_{t-1}, u_t d, 0_{T-t}),$$

where the n-vector d gives the deal cash flow shape. (This cash flow is a $(T+n-1)$-vector.) Thus committing an amount u_t in period t corresponds to a cash flow that starts in period t with amount $u_t d_1$, then in the next period $t+1$, the amount $u_t d_2$, with the last nonzero cash flow amount $u_t d_n$ in period $t+n-1$. The investor's total cash flow is the sum of the cash flows from commitments made in $t = 1, \ldots, T$. We denote this total cash flow as the $(T+n-1)$-vector c.

The deal cash flow shape d typically is negative for the first few entries, which corresponds to the investor putting money into the deal. The remaining entries of d are typically positive, corresponding to the investor receiving income from their earlier investment. The number $\mathbf{1}^T d$ is the undiscounted profit per USD of commitment. This number is typically positive, meaning that the investor will get more cash back than they put in. (Otherwise the investor would not invest.) As an example, consider the deal cash flow shape

$$d = (-0.1, -0.3, -0.2, +0.3, +0.5, +0.2),$$

with $n = 6$. This means that for each USD committed, the investor immediately pays 0.1 USD, then next year 0.3, and in the third year 0.2. These are investments in the deal. In the fourth, fifth, and sixth years the investor receives 0.3, 0.5, and 0.2 USD per USD committed. These are the payoff for the investment. The (undiscounted) profit on this deal is $\mathbf{1}^T d = 0.4$ USD per USD committed.

(a) Explain why the cash flow c and commitments u are related as $c = u * d$. In words: the cash flow is the convolution of the commitments with the cash flow deal shape. You can explain this using the specific case $n = 2$, $T = 3$.

(b) Find the cash flow associated with the commitments

$$u = (1, 2, 3, 3, 3)$$

and the specific deal cash flow shape d given above. (Use `conv()` in Julia, which requires that you have the DSP package installed.) Find the net present value (NPV) of the cash flow at interest rate 8%. (You can read more about NPV on page 22 of the book.)

Remarks. Deals are also called alternative or illiquid investments (since they are not the more usual stocks and bonds, which are easily traded). Examples include venture funds, direct lending, infrastructure investing, private equity, and so on. One very important aspect that we ignore in this problem is that for each deal, the cash flow shape is not known ahead of time, *i.e.*, there is risk when you commit to a deal. Alternative investments (along with the more usual stocks and bonds) can be found in the portfolios of pension funds, insurance funds, and unversity endowments, for example.

7.7 *Circular convolution.* The circular convolution of two n-vectors a, b is the n-vector c defined as

$$c_k = \sum_{i=1}^{n} a_i b_{((k-i) \bmod n)+1}, \quad k = 1, \ldots, n,$$

where $(k - i) \bmod n$ is the remainder of $k - i$ after integer division by n. For example, if $n = 4$,

$$\begin{array}{rcl} c_1 & = & a_1b_1 + a_2b_4 + a_3b_3 + a_4b_2 \\ c_2 & = & a_1b_2 + a_2b_1 + a_3b_4 + a_4b_3 \\ c_3 & = & a_1b_3 + a_2b_2 + a_3b_1 + a_4b_4 \\ c_4 & = & a_1b_4 + a_2b_3 + a_3b_2 + a_4b_1. \end{array}$$

We use the notation $c = a \circledast b$ for circular convolution, to distinguish it from the standard convolution $c = a * b$ defined in the textbook (page 136).

Suppose a is given. Show that $a \circledast b$ is a linear function of b, by giving a matrix $T_{\rm c}(a)$ such that $a \circledast b = T_{\rm c}(a)b$ for all b.

8 Linear equations

8.1 Suppose the 5-vector c gives the coefficients of a quartic (degree four) polynomial $p(x) = c_1 + c_2 x + c_3 x^2 + c_4 x^3 + c_5 x^4$. Express the conditions

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

as a set of linear equations of the form $Ac = b$. Give the sizes of A and b, as well as their entries.

8.2 *Linear function?* The function $f : \mathbf{R}^n \to \mathbf{R}^m$ is linear. Define another function $g : \mathbf{R}^n \to \mathbf{R}$ by $g(x) = (f(x))_1$. Is g a linear function? If so, briefly explain why. If not, give a simple counterexample.

8.3 Which of the following are linear or affine functions $f : \mathbf{R}^2 \to \mathbf{R}^2$? For ones which are linear, express them in the form $f(x) = Ax$ for some specific matrix A. For ones which are affine but not linear, express them in the form $f(x) = Ax + b$ for some specific matrix A and vector b.

(a) $f(x) = 0$

(b) $f(x) = (1, 0)$

(c) $f(x) = (2x_1, 0)$

(d) $f(x) = (-1, x_1 - x_2)$

(e) $f(x) = (x_2, -x_2)$

(f) $f(x) = (x_1^2, x_2^2)$

8.4 Formulate the following problem as a set of linear equations in the form $Ax = b$. Give the numerical values of the elements of the 4×4 matrix A.

Find a polynomial $p(t) = x_1 + x_2 t + x_3 t^2 + x_4 t^3$ that satisfies the four conditions

$$\int_0^1 p(t)dt = b_1, \qquad \int_0^1 tp(t)dt = b_2, \qquad \int_0^1 t^2 p(t)dt = b_3, \qquad \int_0^1 t^3 p(t)dt = b_4.$$

The numbers b_1, b_2, b_3, b_4 on the right-hand sides are given.

9 Linear dynamical systems

9.1 *Predicting growth of US population segments.* Use the US population dynamics model from §9.2 of VMLS to predict the growth of the following population segments over the years 2010–2020, ignoring immigration and emigration, and assuming the birth and death rates remain constant. Plot the results.

- The total US population.
- The total US population age 65 and older.
- The total US population age 6–18 (*i.e.*, school age).
- The total US population age 19–22 (*i.e.*, college age).

(The number ranges are inclusive, for example, 19–22 means 19, 20, 21, and 22.)

Hints.

- For each subpopulation, find a 100-vector c (different for each subpopulation) for which the subpopulation is $c^T x$, where x gives the age distribution. Alternatively, you can use slicing and the sum function, as in `sum(x[a:b])`, with a and b suitably defined.
- See §9.2 of the VMLS Julia companion for how to get the required data, code examples, and some critical Julia hints. To simulate the population dynamics, we suggest following the code segment given in §9.1 of the VMLS Julia companion, which would create a 100×11 matrix X with columns giving the population distribution in 2010, $\ldots$, 2020. Then the 11-vector $X^T c$ gives the predicted subpopulation. (You can also use slicing as mentioned above.)

9.2 *Simulating epidemic dynamics.* In this exercise you will consider the epidemic dynamics example given in §9.3 of the textbook, with various changes to the dynamics matrix. In each case, coefficients of the dynamics matrix not mentioned remain the same as in the original example.

For each of the variations described below, give the associated dynamics matrix A. Then simulate and plot the results over the range $t = 1$ to $t = 210$ (the same range as the example in the textbook). Make a brief comment or two, focussing on the fraction of the population who have died by day $T = 210$, and the maximum over time of the fraction of people who are infected.

(a) *Lower mortality.* Instead of 1% of infected people dying each day, 0.5% do. (To make the column add up to one, we assume that 85.5% of infected people remain infected.)

(b) *Infection control.* Extensive mask wearing and social distancing reduces the infection rate among susceptible people to 3% (from 5%). (The fraction that remains susceptible is then 97%.)

(c) *Decaying immunity.* Each day, 3% of the recovered people become susceptible. (And 97% remain in infection state recovered.)

(d) *Vaccination.* Each day, 1% of the susceptible people are vaccinated, which moves them to infection state recovered (which in this case isn't really a good name for the infection state, since they were never infected). Each day 5% of the susceptible become infected, and 94% remain susceptible.

Hint. The matrix A should always have the property that each of its columns sums to one, or in matrix notation, $A^T\mathbf{1} = \mathbf{1}$. This is because the ith column of A gives the proportions of people in each of the four infection states in the next period, if the whole population has infection state i now.

Julia hint. You'll want to start from the code given in §9.3 of the Julia VMLS companion, which you can you as a good starting point. We suggest that you create a function `sim_plot!(A)` that simulates the epidemic dynamics with the dynamics matrix A and then plots it for $T = 210$ days, starting from the same initial state $x_1 = e_1$, *i.e.*, everyone susceptible. (You need to use the ! to indicate that the function will have a side effect, in this case, displaying a plot.)

Remark. Accurate epidemic dynamics models are not linear dynamical systems, so don't take any of the results too seriously.

9.3 *Cash flows and simple economic models.* In this exercise, you will develop a simple model of an economic system, represented as a dynamical system, to model cash flows and liquidity. For times $t = 1, 2, 3, \ldots$, we will let M_t, S_t, B_t, P_t be the amount of capital held by manufacturers, sellers, banks, and as personal (individual) wealth. Let

$$x_t = \begin{bmatrix} M_t \\ S_t \\ B_t \\ P_t \end{bmatrix}$$

be the vector of total capital in circulation at time t. We will investigate "limiting" circulating capital for different possible transitions of wealth from each sector to other sectors, and you will define the transition matrix A in these scenarios.

(a) Suppose that at time t, manufacturers pay their employees (individuals) 65% of their capital, repay loans to banks with 30% of their capital, and that 5% of the manufacturer capital disappears (say, due to losses, or because it leaves circulation by being saved by other means). Sellers give 50% of their capital to manufacturers (to buy goods to sell), 30% of their capital to banks to repay loans, and 15% of their capital to individual employees as salaries (i.e., personal wealth). Again, 5% of capital is lost. Banks give 30% of their capital to each of manufacturers and sellers, and 35% to individuals, as loans (with a 5% loss rate). Finally individual consumers repay their bank loans with 40% of their capital and buy things from sellers with 55% of their capital, saving 5% (which is removed from circulation). Give the matrix A for which $x_{t+1} = Ax_t$ for this scenario.

(b) Write Julia code to simulate the scenario in part (a) for $T = 100$ steps. Beginning from the initial conditions that $x_1 = e_1, e_2, e_3$, and e_4 (i.e., all capital in one particular group) what does it seem like the circulating capital x_T converges to as T gets large? Plot your result for the initial condition that $x_1 = e_4$, with all capital in the hands of individuals.

(c) Now consider a slightly different scenario, where at each time, a central bank infuses the system with capital by providing the banks with h units of "inflationary" money, that is, we iterate

$$x_{t+1} = Ax_t + b \quad \text{where} \quad b = (0, 0, h, 0).$$

For $h = .1$, repeat the experiment from part (b) for different initial conditions x_1 and $T = 200$. Does it seem that the capital x_T in circulation converges? Does x_1 effect this?

Now we consider a slight variant of the problem in which there are capital holders, whose capital at time t is C_t. (So now $x_t = (M_t, S_t, B_t, P_t, C_t) \in \mathbf{R}^5$.) Assume that the 5% losses in part (a) now all go to capital holders; capital holders give 100% of their capital back to themselves.

(d) Give the transition matrix $A \in \mathbf{R}^{5\times 5}$ for this case.

(e) Repeat your experiments in parts (b) and (c) with your new matrix. What results do you see?

Hint. You'll want to start from the code given in §9.3 of the Julia VMLS companion, which you can you as a good starting point.

10 Matrix multiplication

10.1 *Matrix multiplication Julia timing test.* Determine how long it takes your computer to compute the product of two $n \times n$ matrices for $n = 500, 1000, 2000, 4000$, and use your result for $n = 4000$ to estimate (very crudely) how many Gflops/sec your computer can carry out. (Hopefully your results for the different values of n will give roughly consistent estimates of computer speed.)

The follow code generates two random 500×500 matrices and times the evaluation of their product. (You might run it a few times; the first time might be a bit slower, since the matrix multiplication code has to be loaded and compiled.)

```
A = randn(500,500); B = randn(500,500);
@time C=A*B;
```

How long would it take a person to carry this out, assuming the person can carry out a floating point operation every 10 seconds for 8 hours each day?

10.2 *Customer purchase history matrix.* A store keeps track of its sales of products from K different product categories to N customers over some time period, like one month. (While it doesn't matter for this problem, K might be on the order of 1000 and N might be 100000.) The data is stored in an $N \times K$ matrix C, with C_{ij} being the total dollar purchases of product j by customer i. All the entries of C are nonnegative. The matrix C is typically sparse, *i.e.*, many of its entries are zero.

(a) What is $C\mathbf{1}$?

(b) What is $C^T\mathbf{1}$?

(c) Give a short matrix-vector expression for the total dollar amount of all purchases, by all customers.

(d) What does it mean if $(CC^T)_{kl} = 0$? Your answer should be simple English.

(e) Suppose you run k-means on the rows of C, with $k = 100$. How would you interpret the centroids $z_1, \ldots, z_{100}$?

10.3 *State feedback control.* Consider a time-invariant linear dynamical system with n-vector state x_t and m-vector input u_t, with dynamics

$$x_{t+1} = Ax_t + Bu_t, \quad t = 1, 2, \ldots.$$

The entries of the state often represent deviations of n quantities from their desired values, so $x_t \approx 0$ is a goal in operation of the system. The entries of the input u_t are deviations from the standard or nominal values. For example, in an aircraft model, the states might be the deviation from the desired altitude, climb rate, speed, and angle of attack; the input u_t represents changes in the control surface angles or engine thrust from their normal values.

In *state feedback control*, the states are measured and the input is a linear function of the state, $u_t = Kx_t$. The $m \times n$ matrix K is called the *state feedback gain matrix.* The state feedback gain matrix is very carefully designed, using several methods. State feedback control is very widely used in many application areas (including, for example, control of airplanes).

(a) *Open and closed-loop dynamical system.* With $u_t = 0$, the system satisfies $x_{t+1} = Ax_t$ for $t = 1, 2, \ldots$, which is called the *open-loop dynamics.* When $u_t = Kx_t$, the system dynamics can be expressed as $x_{t+1} = \tilde{A}x_t$, for $t = 1, 2, \ldots$, where the $n \times n$ matrix $\tilde{A}$ is the *closed-loop dynamics matrix.* Find an expression for $\tilde{A}$ in terms of A, B, and K.

(b) *Aircraft control.* The longitudinal dynamics of a 747 flying at 40000 ft at Mach 0.81 is given by

$$A = \begin{bmatrix} .99 & .03 & -.02 & -.32 \\ .01 & .47 & 4.7 & .00 \\ .02 & -.06 & .40 & -.00 \\ .01 & -.04 & .72 & .99 \end{bmatrix}, \qquad B = \begin{bmatrix} 0.01 & 0.99 \\ -3.44 & 1.66 \\ -0.83 & 0.44 \\ -0.47 & 0.25 \end{bmatrix},$$

where the sampling time is one second. (The state and control variables are described in more detail in the lecture on control.) We will use the state feedback matrix

$$K = \begin{bmatrix} -.038 & .021 & .319 & -.270 \\ -.061 & -.004 & -.120 & .007 \end{bmatrix}.$$

(The matrices A, B, and K can be found in `747_cruise_dyn_data.jl`, so you don't have to type them in.) Plot the open-loop and closed-loop state trajectories from several nonzero initial states, such as $x_1 = (1, 0, 0, 0)$, or ones that are randomly generated, from $t = 1$ to $t = 100$ (say). (In other words, plot $(x_t)_i$ versus t, for $i = 1, 2, 3, 4$.) Would you rather be a passenger in the plane with the state feedback control turned off (*i.e.*, open-loop) or on (*i.e.*, closed-loop)?

10.4 *Student-course matrix.* The Stanford registrar has the complete list of courses taken by each graduating student over several graduating classes. This data is represented by an $m \times n$ matrix C, with $C_{ij} = 1$ if student i took class j, and $C_{ij} = 0$ otherwise, for $i = 1, \ldots, m$ and $j = 1, \ldots, n$. (Thus, there are m students in the data set, and n different courses. For simplicity, we ignore the possibility that in some circumstances a student can take a course multiple times.)

Answer each of the questions below in English, with no equations, references to matrices or vectors, and so on. (You can refer to student i and course j, though.)

(a) What is $(C^TC)_{kl}$?

(b) What is $(CC^T)_{rs}$?

(c) What is $(C^T\mathbf{1})_p$?

(d) Suppose you cluster the columns of C^T using k-means, with $k = 50$ (say). What do you think the results might look like? (Your response can be a bit vague, but not more than one or two sentences.)

(e) (Continuation of part (d).) Suppose z_1 is the cluster representative for group 1. What does $(z_1)_{143} = 0.01$ mean?

10.5 Suppose A is a 5×10 matrix, B is a 20×10 matrix, and C is a 10×10 matrix. Determine whether each of the following expressions make sense. If the expression makes sense, give its dimensions.

(a) $A^TA + C$.

(b) BC^3.

(c) $I + BC^T$.

(d) $B^T - [C\ I]$.

(e) $B \left[\begin{array}{c} A \\ A \end{array} \right] C$.

10.6 *Friend matrix.* We consider a collection of n people who participate in a social network in which pairs of people can be connected, by 'friending' each other. The $n \times n$ matrix F is the friend matrix, defined by $F_{ij} = 1$ if persons i and j are friends, and $F_{ij} = 0$ if not. We assume that the friend relationship is symmetric, *i.e.*, person i and person j are friends means person j and person i are friends. We will also assume that $F_{ii} = 0$. Express the following in matrix/vector notation, briefly justifying your expression.

(a) t is the n-vector with t_i being the total number of friends of person i.

(b) C is the $n \times n$ matrix with C_{ij} equal to the number of friends persons i and j have in common. (Person k is a friend in common of persons i and j if she is a friend of both person i and person j. The diagonal entry C_{ii}, which is the total number of friends person i has in common with herself, is the total number of friends of person i.)

10.7 Suppose $F^T G = 0$, where F and G are $n \times k$ matrices. Determine whether each of the following statements must always be true, or can be false. 'Must be true' means the statement holds for any $n \times k$ matrices F and G that satisfy $F^T G = 0$, without any further assumptions; 'can be false' means that there are $n \times k$ matrices F and G that satisfy $F^T G = 0$, but the statement does not hold.

(a) Either $F = 0$ or $G = 0$.

(b) The columns of F are orthonormal.

(c) Each column of F is orthogonal to each column of G.

(d) The matrices F and G are square or tall, *i.e.*, $n \geq k$.

(e) The columns of F are linearly dependent.

10.8 *Matrix with acute columns.* Suppose A is an $m \times n$ matrix with nonzero columns $a_1, \ldots, a_n$, and in addition, any pair of columns makes an acute angle, *i.e.*, $|\angle(a_i, a_j)| < 90^\circ$ for all $i, j = 1, \ldots, n$. What can you say about the entries of the Gram matrix $G = A^T A$?

10.9 Given n-vectors a, b. Which of the following are true?

(a) $ab^T = ba^T$

(b) $ab^T = (ba^T)^T$

(c) $ab^T = b^T a$

(d) $a^T b = b^T a$

(e) $ab^T = a^T b$

10.10 Suppose A is an $m \times n$ matrix. Express the sum of all entries of A using matrix and vector notation. Your solution can include any matrix and vector operations (sum, product, scalar multiplication, transpose, ...), unit vectors, zero vectors, and ones vectors. If you use zeros or ones vectors, you must given their dimensions. We will deduct points from correct answers that are more complicated than they need to be. Recall that we consider a 1×1 matrix to be the same as a number, so your solution can be a 1×1 matrix.

10.11 *Square of sum.* If A and B are square matrices with the same dimensions, then $(A+B)^2 =$

(a) $A^2 + 2AB + B^2$

(b) $A^2 + 2BA + B^2$

(c) $A^2 + AB + BA + B^2$

(d) all of the above

(e) none of the above

(Choose one.)

10.12 *Time to compute the matrix square.* A particular computer takes around one second to compute the square of a 3000×3000 matrix. About how long will it take to compute the square of a 300×300 matrix?

10.13 *QR factorization of matrix with orthonormal columns.* Suppose the columns of a matrix A are orthonormal, and we (attempt) to compute its QR factorization $A = QR$. Which of the following must be true?

(a) The QR factorization will fail.

(b) $R = I$

(c) $R = A$

(d) $Q = I$

(e) $Q = A$

10.14 *Total economic output in three years.* The n-vector x_t denotes the economic output of a country in n different sectors, in year t. An economist models the economic output using a linear dynamical system model $x_{t+1} = Ax_t$, where A is an $n \times n$ matrix. Assuming her model is correct, find an expression for the total economic output in year $t+3$, in terms of x_t and A. (The total economic output is the sum of the economic output in the n sectors.)

10.15 The $n \times K$ data matrix X describes whether each of K items has each of n Boolean attributes: $X_{ij} = 1$ means item j has attribute i; otherwise, $X_{ij} = 0$, which means item j does not have attribute i. Suppose p and q are integers between 1 and K, and $p \neq q$. Express $(X^T X)_{pq}$ in simple English.

10.16 *Diagonal pre- and post multiplication.* Suppose A is an $m \times n$ matrix, D is a diagonal $m \times m$ matrix, and E is a diagonal $n \times n$ matrix. Let $C = DAE$.

What is C_{ij}, in terms of the entries of A, D, and E? We will deduct points for correct answers that are more complicated than they need to be.

10.17 *Matrix dimensions.* Suppose A is an $m \times n$ matrix, and B is a $p \times q$ matrix. For each of the following equations, give the conditions on m, n, p, q under which it makes sense. For example, $A = B$ requires that $m = p$ and $n = q$.

(a) $A^TB = I$.

(b) $AB = 0$.

(c) $AB = BA$.

(d) $A^T = [B\ I]$.

(e) $[A\ B] = I$.

10.18 *Number of paths of a given length in a directed graph.* Consider a directed graph with n nodes, specified by its $n \times n$ adjacency matrix A (see textbook, page 112). What is N, the total number of paths of length 10 in the graph? (See textbook, §10.3). Give an expression for N that involves the matrix A and standard matrix operations like matrix-matrix multiplication, matrix powers, transpose, slicing, matrix-vector multiplication, and standard vectors like 0 and $\mathbf{1}$. Your answer should use only matrix and vector notation.

10.19 *The adjoint identity.* Suppose A is an $m \times n$ matrix. Show that for any m-vector u and n-vector v we have $u^T(Av) = (A^Tu)^Tv$. In words: The inner product of u and Av is the same as the inner product of A^Tu and v. (Note that these inner products are of vector with different dimensions.) This equation is called the *adjoint identity.* (Adjoint is another name for transpose, *i.e.*, A^T is called the adjoint of A.)

10.20 *Quadratic form.* Suppose P is an $n \times n$ matrix. The function $f : \mathbf{R}^n \rightarrow \mathbf{R}$ defined as $f(x) = x^TPx$ is called a *quadratic form*, and generalizes the idea of a quadratic function of a scalar variable, px^2. The matrix P is called the coefficient matrix of the quadratic form.

(a) Show that $f(x) = \sum_{i,j=1}^{N} P_{ij}x_ix_j$. In words: $f(x)$ is the weighted sum of all products of two components of x, with weights given by the entries of P.

(b) Show that for any x, we also have $f(x) = x^T(P^T)x$. In other words, the quadratic form associated with the transpose matrix is the same function. *Hint.* Take the transpose of $f(x) = x^TPx$.

(c) Show that f can be expressed as $f(x) = x^TP^{\text{sym}}x$, where $P^{\text{symm}} = (1/2)(P + P^T)$ is the *symmetric part* of P. The matrix P^{sym} is symmetric. So any quadratic form can be expressed as one with a coefficient matrix that is symmetric.

(d) Express $f(x) = -2x_1^2 + 4x_1x_2 + 2x_2^2$ in the form $f(x) = x^TPx$ with P a symmetric 2×2 matrix.

(e) Suppose that A is an $m \times n$ matrix and b is an m-vector. Show that $\|Ax - b\|^2 = x^TPx + q^Tx + r$ for a suitable $n \times n$ symmetric matrix P, n-vector q, and constant r. (Give P, q, and r.) In words: The norm squared of an affine function of x can be expressed as the sum of a quadratic form and an affine function.

10.21 *Block matrices.* Generate four random 3 by 5 matrices A, B, C, and D.

(a) Verify that

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^T = \begin{bmatrix} A^T & B^T \\ C^T & D^T \end{bmatrix}.$$

(b) Generate four more random 3 by 5 matrices E, F, G, and H. Verify that

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}\begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}.$$

You can do this evaluating the lefthand and righthand sides, and computing the norm of the difference, which should be a really small number.

10.22 Let A be a lower triangular $n \times n$ matrix. Verify the following properties.

(a) If B is a lower triangular $n \times n$ matrix, then the product AB is lower triangular.

(b) The matrix A^k is lower triangular for all positive integers k.

(c) If A is invertible, then A^k is lower triangular for all integers k (positive or negative).

10.23 Consider a ball on a flat track, where at time t the ball has velocity v_t. We approximate this velocity by the linear dynamical system evolving over time steps $t = 1, 2, \ldots$ as

$$v_{t+1} = av_t + bu_{t+1}, \tag{1}$$

where $0 < a < 1$ is the acceleration (really, deceleration) from friction, u_{t+1} is a force applied from t through $t + 1$ (we use this $t + 1$ indexing to simplify the math later, as usually we would write u_t instead of u_{t+1}). This force is also known as the control action. The mass of the ball is $1/b$.

(a) Give the impulse response vector h for this system through time k. That is, give the vector $h \in \mathbf{R}^k$ such that if $u_1 = 1$ (meaning we apply a unit positive force at time 0) and $v_0 = 0$, we have

$$h_t = v_t \quad \text{for} \quad t = 1, 2, \ldots, k.$$

(b) *A finite impulse response.* Make the simplifying assumption that if we apply a force u_{t+1} at a time t to the ball, its effect on the ball's velocity follows the dynamics (1) for k steps and then disappears. That is, assuming $u_{t+1} \neq 0$ while $u_{t+2} = \cdots = u_{t+k+1} = 0$, we have

$$v_{t+1} = av_t + bu_{t+1}, \quad v_{t+2} = av_{t+1}, \quad \ldots, \quad v_{t+k} = av_{t+k-1}, \quad v_{t+k+1} = 0.$$

Give a vector $h \in \mathbf{R}^k$ such that for any $T \in \{1, 2, \ldots\}$ and vector $u \in \mathbf{R}^T$ of forces applied at times $t = 0, 1, \ldots, T - 1$, the velocities $v \in \mathbf{R}^{k+T-1}$ from times $t = 1$ through $t = k + T - 1$ satisfy

$$v = h * u.$$

(c) Keeping the notation and simplified dynamics of part (b), we are interested in the position x_t of the ball at time t when we apply a sequence of forces $u \in \mathbf{R}^T$. Assume that $x_0 = 0$ and $v_0 = 0$ so the ball starts at rest and its initial position is 0, and that if the ball has velocities $v_1, v_2, \ldots, v_t$, then its position at time t is

$$x_t = x_0 + v_0 + v_1 + \cdots + v_{t-1}.$$

(That is, in each time step t, the ball moves distance v_t). Give a matrix A such that the vector $x \in \mathbf{R}^T$ of positions of the ball at times $t = 1, 2, \ldots, T$ satisfies $x = A(h * u)$ given any vector $u \in \mathbf{R}^T$ of forces, where h is your vector from part (b). Specify the dimensions of A and its entries.

10.24 *Householder transforms and QR factorization.* For a unit vector u, i.e., vector with $\|u\| = 1$, define the matrix $H_u = 2uu^T - I$

(a) Give H_u^2.

(b) Let $u_1, \ldots, u_k$ be any unit n-vectors. Show that the product

$$Q = H_{u_1} H_{u_2} \cdots H_{u_k}$$

is an $n \times n$ orthogonal matrix, that is, $Q^T Q = QQ^T = I$.

(c) Let

$$A = [a_1 \ \cdots \ a_n] \in \mathbf{R}^{m \times n}$$

have non-zero first column $a_1 \in \mathbf{R}^m$, and assume that its first entry $a_{11} \geq 0$. Give a unit vector u so that

$$H_u A = \begin{bmatrix} \|a_1\| & b^T \\ \mathbf{0}_{n-1} & C \end{bmatrix},$$

where $\mathbf{0}_{n-1}$ is the $(n-1)$-vector of all zeros, and $b \in \mathbf{R}^{n-1}$ and $C \in \mathbf{R}^{m-1 \times n-1}$ are a vector and matrix of appropriate sizes. (You do not need to specify them.) *Hint.* See problem 6.15.

10.25 We have implemented code to compute QR factorizations using Householder transformations, which is available at `https://web.stanford.edu/class/engr108/julia_files/householder.jl`. The method `compute_QR` takes a matrix as an argument and returns a pair `(Q, R)`, where to within numerical tolerance, it should be the case that `Q * R = A`. In this question, you will investigate this method. Recall that the Householder method to compute the QR factorization gives

$$Q = H_{u_1} H_{u_2} \cdots H_{u_m},$$

where u_i is a unit vector whose first $(i-1)$ entries are 0, and $H_{u_i} = I - 2u_i u_i^T$ is a Householder transform matrix, while $A = QR$ for an upper triangular R.

In this question, let $n = 8$, and the matrix $A \in \mathbf{R}^{n \times k}$, where k will vary.

(a) For $k = 3$, generate a random $n \times k$ matrix A and let (Q, R) be the computed factorization of A. Verify (in Julia) that `norm(Q * R - A)` is small. What structure does `R` appear to have?

(b) Again for $k = 3$, generate a random $n \times k$ matrix A, but set is third column equal to its first. Repeat part (a) for this matrix.

(c) Repeat part (a), except set $k = 12$.

(d) Repeat part (a), except set the last four columns of `A` to be identical to the first four columns of `A`.

(e) Let `u = find_annihilator(A)` for (any) one of your matrices above. Evaluate $(I - 2uu^T)A$ and $(2uu^T - I)A$ for this u and A. Explain the difference in your results.

10.26 Let A be an $n \times n$ matrix and let $QR = A$ be a factorization of A where $Q \in \mathbf{R}^{n \times n}$ is orthogonal matrix and $R \in \mathbf{R}^{n \times n}$ is upper triangular.

(a) Give a diagonal matrix D such that the diagonal of DR is nonnegative and

$$D^2 = I.$$

(b) For your matrix, does QD have orthonormal columns?

(c) Given a $A = QR$ where R may have non-positive entries on its diagonal, explain how to construct an orthogonal matrix $\widetilde{Q}$ and upper triangular matrix $\widetilde{R}$ with nonnegative diagonal and for which $A = \widetilde{Q}\widetilde{R}$.

10.27 Let A be an $n \times n$ matrix with nonnegative entries, and let its columns $a_i \in \mathbf{R}^n_+$ satisfy $\mathbf{1}^T a_i < 1$.

(a) Why is there a $0 \leq \beta < 1$ such that each entry of the vector $v = A^T\mathbf{1}$ satisfies $v_i \leq \beta$?

(b) Are all (positive) powers of A nonnegative? That is, does A^k have nonnegative entries for all $k = 1, 2, \ldots$?

(c) Justify that for each $k = 1, 2, \ldots,$

$$\mathbf{1}^T(A^k)\mathbf{1} \leq \beta^k n^2.$$

(In fact, one can show $\mathbf{1}^T(A^k)\mathbf{1} \leq \beta^k n$, but this isn't necessary.)

(d) In the linear dynamical system

$$x_{k+1} = Ax_k,$$

does x_k converge as $k \to \infty$? If so, to what?

11 Matrix inverses

11.1 *Rows and columns of a matrix and its inverse.* Suppose the $n \times n$ matrix A is invertible, with inverse $B = A^{-1}$. We let the n-vectors $a_1, \ldots, a_n$ denote the columns of A, and $b_1^T, \ldots, b_n^T$ the rows of B. Determine whether each of the following statements is true or false, and justify your answer. *True* means the statement always holds, with no further assumptions. *False* means the statement does not always hold, without further assumptions.

(a) For any n-vector x, we have $x = \sum_{i=1}^n (b_i^T x) a_i$.

(b) For any n-vector x, we have $x = \sum_{i=1}^n (a_i^T x) b_i$.

(c) For $i \neq j$, $a_i \perp b_j$.

(d) For any i, $\|b_i\| \geq 1/\|a_i\|$.

(e) For any i and j, $b_i + b_j \neq 0$.

(f) For any i, $a_i + b_i \neq 0$.

11.2 *Solving linear equations in Julia.* Generate a random 20×20 matrix A and a random 20-vector b using the following code:

```
A = rand(20, 20)
b = rand(20)
```

(It's very likely that the matrix you generate will be invertible.)

We solve the linear equation $Ax = b$, *i.e.*, compute the solution $x = A^{-1}b$ in Julia using several methods. In each case, you should check that the x you compute satisfies the equations by evaluating and reporting the norm of the residual, $\|Ax - b\|$. (This should be very small.)

(a) Using the backslash operator:

```
x = A \ b
```

(b) Computing the inverse of A explicitly:

```
x = inv(A) * b
```

(c) Using QR factorization, from the formula $x = R^{-1}Q^T b$:

```
Q, R = qr(A)
x = R \ (Q' * b)
```

(You should check that the matrix Q obtained is very nearly orthogonal, R is an upper triangular matrix, and that A is very near QR.)

11.3 *Julia timing test for linear equations.*

(a) Determine how long it takes for your computer to solve a system of $n = 2000$ linear equations in $n = 2000$ variables (with invertible coefficient matrix) using Julia's \ operator. You may use the following code.

```
A = 1 + rand(2000, 2000)
b = ones(2000)
@time A\b;
```

(b) Julia is rather clever about how it solves systems of equations with \. Determine how long it takes for your computer to solve the following system of $n = 2000$ linear equations in $n = 2000$ variables.

```
L = 1 + rand(2000, 2000)
for i = 1:2000
  for j = i+1:2000
    L[i, j] = 0
  end
end
b = ones(2000)
@time L\b;
```

(c) Can you explain why the times differ by so much between the two systems, *i.e.*, what is special about the matrix L as opposed to A? Make a hypothesis about what you think Julia is doing behind the scenes.

11.4 *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 now perturb the jth entry b by $\epsilon \neq 0$ (which is a traditional symbol for a small quantity), which means that 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.

(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 dervied from A) and ϵ only.

(c) Try this out in Julia with the following values:

$$A = \begin{bmatrix} 1/2 & 1/3 & 1/4 \\ 1/3 & 1/4 & 1/5 \\ 1/4 & 1/5 & 1/6 \end{bmatrix}, \qquad b = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \qquad \epsilon = 0.1.$$

To prevent numerical imprecision errors, use the following code to build the data.

```
A = [1/2 1/3 1/4; 1/3 1/4 1/5; 1/4 1/5 1/6]
b = [1.0, 1.0, 1.0]
epsilon = 0.1
```

Which j do you pick to maximize $\|x - \tilde{x}\|$, and what value do you get for $\|x - \tilde{x}\|$? Check your answer from part (b) by direct calculation (*i.e.*, simply finding $\tilde{x}$ after perturbing entry $j = 1, 2, 3$ of b).

11.5 Let A and B be distinct, square $n \times n$ matrices that satisfy $A^5 = B^5$ and $AB^4 = BA^4$. Determine whether the matrix $A^4 + B^4$ is invertible. If it is, give an explicit formula to compute its inverse. If it is not, provide an explanation.

Hint: Recall that matrix multiplication distributes over addition.

11.6 *Julia for circuit analysis.* This problem relates to material from the additional lecture on electrical circuit analysis, at `https://stanford.edu/class/engr108/lectures/circuits.pdf`.

(a) Write Julia code that solves for the branch currents and node potentials in the circuit described on slide 20, and verify the results we report on slide 21.

(b) Write Julia code that finds the resistance matrix $\mathcal{R}$ for the circuit given on slide 24, and verify that it agrees with what we report at the bottom of that slide.

(c) Write Julia code that checks conservation of power, *i.e.*, evaluates $P^{\text{diss}} = j^T v$ and $P^{\text{ext}} = i^T e$, for the specific example, *i.e.*, the solution of part (b). (They should be the same, except for small rounding errors.)

Hints.

- First define the important data for the problems, *i.e.*, the matrices A and R (for parts (a) and (b)), and the vector i (for part (a)).
- Then write generic code that finds j and e (for part (a)), or $\mathcal{R}$ (for part (b)), using only these data. (So your code would actually work for *any* circuit.)
- To form a stacked vector in Julia from `a` and `b`, use `[a ; b]`.
- To form a 2×2 block matrix in Julia, with subblocks `A`, `B`, `C`, and `D`, use `[A B; C D]`.
- Use slicing in Julia to extract a subvector from a bigger vector. For example, to get the last n components of a $(b+n)$-vector `z`, use `e = z[b+1:b+n]`.

11.7 Let $a = \begin{bmatrix} 1 & -1 \end{bmatrix}$. Does a have a left inverse? A right inverse? If not, say so. If so, give one.

11.8 For each statement below, say whether it is true or false. If it is false, give a counter-example.

(a) If a square matrix A has a left inverse, then it has a right inverse.

(b) If a matrix has a left and a right inverse, it must be square.

11.9 *Inverse of square.* Suppose A is an invertible $n \times n$ matrix, with inverse A^{-1}. Is A^2 invertible? If so, given an expression for its inverse (in terms of A and its inverse). If not, give a specific counter-example of a matrix that is invertible, but its square is not. You do not need to justify your counter-example, but please do keep it as simple as possible.

11.10 *Deducing a cash flow from its net present value under different interest rates.* The net present value (NPV) of a cash flow $c = (c_1, c_2, c_3)$ with interest rate r is $c_1 + c_2/(1+r) + c_3/(1+r)^2$. With interest rate $r = 0$, the NPV is 1. With interest rate $r = 0.05$, the NPV is 0. With interest rate $r = 0.10$, the NPV is -1.

Find the cash flow c. You are welcome, indeed encouraged, to use a computer to solve any equations you might need to.

11.11 Find two different right inverses of the 2×3 matrix $A = \begin{bmatrix} 1 & -2 & 0 \\ 0 & 1 & 2 \end{bmatrix}$. *Hints.* We suggest you solve this by hand. (That's a suggestion we very rarely make.) You are free to use a computer to check your answers. There are of course multiple correct answers.

11.12 *Inverse of matrix with positive entries.* If a number a is positive, so is its inverse $1/a$. Is the same true for matrices? That is, if an invertible matrix A has all its entries positive, are all the entries of its inverse positive?

11.13 *Polynomial interpolation with derivatives.* The 4-vector c contains the coefficients of a cubic polynomial $p(x) = c_1 + c_2x + c_3x^2 + c_4x^3$.

(a) Express the following conditions on p as a set of linear equations for the coefficients, *i.e.*, $Ac = b$. You must give A and b. The conditions are

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

(b) Solve the equations you set up in part (a), and plot p over the range $-0.1 \leq x \leq 1.1$, checking that it satisfies the conditions. These equations are readily solved by hand, but feel free to use Julia instead.

11.14 *Two trajectories of a linear dynamical system.* The n-vectors $x_1, \ldots, x_N$, with $N > 1$, satisfy the time-invariant linear dynamics $x_{t+1} = Ax_t$ for $t = 1, \ldots, N-1$, and the n-vectors $z_1, \ldots, z_N$ satisfy the same time-invariant linear dynamics $z_{t+1} = Az_t$ for $t = 1, \ldots, N-1$, where A is the $n \times n$ dynamics matrix.

Which of the following two statements are true, meaning that they always hold, with no other conditions imposed?

(a) If $x_1 = z_1$, then $x_N = z_N$. In words: if the two trajectories start in the same state, they will end in the same state.

(b) If $x_1 \neq z_1$, then $x_N \neq z_N$. In words: if the two trajectories start in different states, they will end in different states.

We now add the further condition that A is invertible. With this additional assumption, are the two statements above true?

All together you will give four answers, each True or False, for the two statements above, and without and with the additional assumption that A is invertible. You do not need to justify your answers.

Hint. The statement (b) is the same as saying that if $x_N = z_N$, then $x_1 = z_1$.

11.15 *Non-invertibility of a tall-wide matrix product.* Suppose that A is an $m \times n$ matrix, and B is an $n \times m$ matrix, so $C = AB$ is an $m \times m$ matrix. We assume that $m > n$, so A is tall and B is wide. We refer to C as a *tall-wide matrix product.*

Explain why the matrix C is not invertible. In words: a tall-wide matrix product is never invertible.

Hint. Argue by contradiction: first assume that C is invertible, and then derive a contradiction. You can flesh out the steps below.

- If C is invertible, there is an $m \times m$ matrix X for which $XC = I$.
- It follows that XA is a left inverse of B.
- We know that the wide matrix B cannot have a left inverse, so we have our contradicition.

11.16 *Deducing system state from component measurements at different times.* Consider a linear dynamical system $x_{t+1} = Ax_t$, $t = 1, \ldots, T$, with state dimension n. We measure (*i.e.*, know) the n numbers $(x_1)_1, (x_2)_2, \ldots, (x_n)_n$, *i.e.*, ith component of the state in period i, for $i = 1, \ldots, n$.

Find the $n \times n$ matrix F for which $((x_1)_1, \ldots, (x_n)_n) = Fx_1$.

The matrix F depends on A. You can describe F by giving its rows or columns. You can make use of standard concepts like slicing, rows and columns, unit vectors, matrix powers, transpose, and so on.

Remark. If F is invertible, this allows us to find the initial state x_1 from the component measurements $(x_1)_1, (x_2)_2, \ldots, (x_n)_n$.

11.17 Define a sequence of matrices A_k, for $k = 0, 1, 2\ldots$, as follows: $A_0 = 1$ and for $k \geq 1$,

$$A_k = \left[\begin{array}{rr} A_{k-1} & A_{k-1} \\ -A_{k-1} & A_{k-1} \end{array}\right].$$

Therefore A_k is a matrix of size $2^k \times 2^k$. The first four matrices in the sequence are

$$A_0 = 1, \qquad A_1 = \left[\begin{array}{rr} 1 & 1 \\ -1 & 1 \end{array}\right], \qquad A_2 = \left[\begin{array}{rrrr} 1 & 1 & 1 & 1 \\ -1 & 1 & -1 & 1 \\ -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 \end{array}\right],$$

$$A_3 = \left[\begin{array}{rrrrrrrr} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\ -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 \end{array}\right].$$

Show that the matrices A_k are invertible. What is the inverse of A_k?

Hint. Note that the columns of A_k are mutually orthogonal and have norm $\sqrt{n}$ where $n = 2^k$.

11.18 (a) Formulate the following problem as a set of linear equations. Find a point $x \in \mathbf{R}^n$ at equal distance to $n+1$ given points $y_1, y_2, \ldots, y_{n+1} \in \mathbf{R}^n$:

$$\|x - y_1\| = \|x - y_2\| = \cdots = \|x - y_{n+1}\|.$$

Write the equations in matrix form $Ax = b$.

(b) Show that the solution x in part (a) is unique if the $(n+1) \times (n+1)$ matrix

$$\left[\begin{array}{cccc} y_1 & y_2 & \cdots & y_{n+1} \\ 1 & 1 & \cdots & 1 \end{array}\right]$$

is invertible.

11.19 Let A be an $n \times n$ matrix with the property that

$$|A_{ii}| > \sum_{j \neq i} |A_{ij}|, \quad i = 1, \ldots, n.$$

(The absolute value of each diagonal element is greater than the sum of the absolute values of the other elements in the same row.) Such a matrix is called *strictly diagonally dominant*. Show that A is invertible.

Hint. Consider a vector x with $Ax = 0$. Suppose i is an index with $|x_i| = \max_{j=1,\ldots,n} |x_j|$. Consider the ith equation in $Ax = 0$.

11.20 If A and B are invertible matrices of the same size, then AB is invertible with inverse $(AB)^{-1} = B^{-1}A^{-1}$. It is tempting to generalize this to

$$(AB)^{\dagger} = B^{\dagger}A^{\dagger} \tag{2}$$

for matrices A and B with linearly independent columns.

(a) First show that (2) is false for

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

We conclude that (2) does not necessarily hold for matrices with linearly independent columns.

(b) Next, we make stronger assumptions on A and B. In each of the following two subproblems, either show that (2) holds, or give a small example for which $(AB)^{\dagger} \neq B^{\dagger}A^{\dagger}$. We assume the dimensions of A and B are compatible, so the product AB is defined.

(i) A has linearly independent columns and B is invertible.

(ii) A is invertible and B has linearly independent columns.

11.21 A square matrix A is called *normal* if $AA^T = A^TA$. Show that if A is normal and invertible, then the matrix $Q = A^{-1}A^T$ is orthogonal.

11.22 When is a matrix lower-triangular and orthogonal? How many $n \times n$ matrices with this property exist?

11.23 Suppose you have to solve two sets of linear equations

$$Ax_1 = b_1, \qquad A^Tx_2 = b_2$$

where A is $n \times n$, b_1 and b_2 are n-vectors, and A is invertible. The unknowns are the n-vectors x_1 and x_2. Describe a method for solving the two equations with a single matrix factorization. What is the complexity of the method?

11.24 Suppose that Q is an $n \times n$ matrix with orthonormal columns. Does Q have orthonormal rows? Why or why not?

11.25 Let A be an $n \times n$ matrix.

(a) Show that for $k = 0, 1, 2, \ldots,$

$$(I - A)(I + A + A^2 + \cdots + A^k) = I - A^{k+1}.$$

(b) Now suppose that A has nonnegative entries, and let a_i be its ith column. Assume $a_i^T \mathbf{1} \leq \beta < 1$ for each $i = 1, \ldots, n$, where β is a (fixed) scalar. Give a formula for $(I - A)^{-1}$ involving sums of powers of A. *Hint.* Recall Question 10.27.

(c) Consider a matrix A as in part (b), and consider the linear dynamical system

$$x_{k+1} = Ax_k + b, \quad k = 0, 1, 2, \ldots,$$

where $b \in \mathbf{R}^n$. Why does x_k have the limit $x_\infty = (I - A)^{-1}b$?

(d) Suppose we wish to guarantee that $x_k \to x_\infty$ for some vector $x_\infty \in \mathbf{R}^n$. How should we choose b to guarantee this?

11.26 *Controlling a positive LDS.* Let A be the matrix from part (a) of problem 9.3. Use Julia to give the vector b guaranteeing the iterates x_k converge to the following vectors:

(a) $x_\infty = \lim_{k\to\infty} x_k = (.25, .25, .25, .25)$

(b) $x_\infty = \lim_{k\to\infty} x_k = (.1, .2, .3, .4)$

(c) For each of the two cases above, write (or re-use) Julia code to simulate the dynamical system in problem 9.3(a), beginning from the initial condition $x_0 = e_4$, the fourth standard basis vector. Give (separate) plots of the trajectory of your solution for $T = 200$ steps.

11.27 *Block matrix inversion and the Woodbury formula.* Consider a block matrix

$$\begin{bmatrix} A & U \\ V^T & -C^{-1} \end{bmatrix}$$

where A, U, V, C have appropriate sizes, and we assume both A and C are invertible.

(a) Show that if X and Y solve

$$\begin{bmatrix} A & U \\ V^T & -C^{-1} \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} I \\ 0 \end{bmatrix},$$

then $(A + UCV^T)X = I$, that is, $X = (A + UCV^T)^{-1}$.

(b) Solve for X explicitly in part (a) to show that

$$(A + UCV^T)^{-1} = A^{-1} - A^{-1}U(V^TA^{-1}U + C^{-1})^{-1}V^TA^{-1}.$$

This is known as the *Woodbury matrix identity.*

(c) Now let $A \in \mathbf{R}^{m\times n}$ be a matrix whose ith row is a_i^T, where $a_i \in \mathbf{R}^n$ is an n-vector. You may assume A^TA is invertible, and that $0 \leq a_i^T(A^TA)^{-1}a_i < 1$ for each $i = 1, \ldots, m$. Use the Woodbury matrix identity to show that

$$\left(A^TA - a_ia_i^T\right)^{-1} = (A^TA)^{-1} - \frac{(A^TA)^{-1}a_ia_i^T(A^TA)^{-1}}{a_i^T(A^TA)^{-1}a_i - 1}.$$

12 Least squares

12.1 *Solving least squares problems in Julia.* Generate a random 20×10 matrix A and a random 20-vector b.

(a) Compute the solution $\hat{x}$ of the associated least squares problem using the methods listed below, and verify that the solutions found are the same, or more accurately, very close to each other; they will be very slightly different due to small roundoff errors in the computations.

- Using the Julia backslash operator.
- Using $\hat{x} = (A^T A)^{-1} A^T b$.
- Using $\hat{x} = A^\dagger b$.

Hints. In Julia, `inv()` computes the inverse matrix, `pinv()` computes the pseudo-inverse matrix, and `A\b` directly solves the least squares problem.

(b) Let $\hat{x}$ be one of the solutions found in part (a). Generate a random nonzero 10-vector δ and verify that $\|A(\hat{x}+\delta) - b\|^2 > \|A\hat{x} - b\|^2$. Repeat several times with different values of δ; you might try choosing a small δ (say, by scaling the original random vector).

Be sure to submit your code, including the code that checks if the solutions in part (a) are close to each other, and whether the expected inequality in part (b) holds.

12.2 *Julia timing test for least squares.* Determine how long it takes for your computer to solve a least squares problem with $m = 100000$ equations and $n = 100$ variables. (You can use the backslash operator.)

Remark. Julia compiles just in time, so you should run the code a few times to get the correct time.

12.3 *Active noise cancellation.* Active noise cancellation is used to reduce the noise level heard at a specific reference location. This is done using several microphones that pick up ambient noise, a loudspeaker near the reference loction, and a microphone that monitors the sound at the reference location. The signals from the ambient noise microphones are combined, added to the desired signal, and played through the loudspeaker in such a way that the noise heard at the reference location is cancelled, or at least reduced. In real active noise cancellation systems the noise microphone signals are convolved with filters before being added, but we will consider here a simplified setup in which a simple linear combination of them is played through the speaker. We will also assume here that the speaker is perfect.

Here is a mathematical description of the method. We let the scalar time series $y_1, y_2, \ldots$ denote the desired signal, which we know. We let the scalar time series $c_1, c_2, \ldots$ denote the noise cancellation signal, which we will create. We play the signal $y_t + c_t$ through the loudspeaker. At the reference location we hear $y_t + c_t + n_t$, where $n_1, n_2, \ldots$ is a noise signal that we do not know. If we can choose c_t so that $c_t + n_t \approx 0$, then we have achieved (approximate) noise cancellation.

The K ambient noise signals are given by the K-vector time series $a_1, a_2, \ldots$. We use $c_t = \theta^T a_t$, where the K-vector θ gives the coefficients for combining the ambient noise microphone signals to form the noise cancellation signal. Our goal is to choose θ so that $\theta^T a_t + n_t \approx 0$.

Here is how we will choose θ. We use a training period, say, $t = 1, \ldots, T$. We send nothing (*i.e.*, the zero signal) to the loudspeaker, and record the noise $n_1^{\text{train}}, \ldots, n_T^{\text{train}}$ using the reference microphone. We also record the ambient noise signals $a_1^{\text{train}}, \ldots, a_T^{\text{train}}$. We then choose θ to minimize $\sum_{t=1}^{T}(\theta^T a_t^{\text{train}} + n_t^{\text{train}})^2$.

Once we have $\hat{\theta}$ we are ready to deploy noise cancellation. We send $y_t + \hat{\theta}^T a_t$ to the loudspeaker, and we hear $y_t + \hat{\theta}^T a_t + n_t$ at the reference location.

(In real active noise cancellation systems, the weights are updated continuously, while the desired signal is playing.)

The file `anc_data.jl` contains data for an active noise cancellation problem. When you execute this file it will include the audio files. It also contains some helper code for writing out a vector as an audio file, which you can then listen to. You will find the following signals:

- For training, a T-vector `n` and the $K \times T$ matrix `a`, which you will use to determine θ. Be sure to listen to the noise signal, and the individual rows of `a`, which give the ambient noise microphone signals. You can check your active noise cancellation by forming and then listening to $n_t + \hat{\theta}^T a_t$. (This should sound quieter than n_t, which is what you would hear without active noise cancellation.)
- For testing, we give you some test signals `y_plus_n_test`, which gives $y_t + n_t$ (over the test interval), and `a_test`, a matrix which gives a_t (over the test interval). To test your $\hat{\theta}$, you should form and listen to $y_t + \hat{\theta}^T a_t + n_t$. Compare this to what $y_t + n_t$ sounds like.

12.4 *Transit system tomography.* An urban transit system (say, a subway) consists of n links between pairs of stations. Each passenger enters the system at an origin station, traverses over a sequence of the links, and then exits the system at their destination station. The fare collection system keeps track of the following information for passenger i: s_i, the trip starting time (when she enters the origin station), measured in minutes after 6AM, f_i, the trip finishing time (when she leaves the destination station), and the sequence of links over which she traveled. For example, $s_i = 128$, $f_i = 144$, and link list $(3, 7, 8, 10, 4)$ means the passenger entered the origin station at 8:08AM, left the destination station at 8:24AM, and traversed links $3, 7, 8, 10, 4$, in that order. The total trip time is $f_i - s_i$.

We model the trip time as the sum of the delays over the links the passenger traverses. We let d_i denote the delay associated with link i, for $i = 1, \ldots, n$. We do not know the link delays, but wish to estimate them, based on the passenger information described above, for a very large number m of passengers passing through the system.

We will do this using least-squares. We choose our estimate $\hat{d}$ of the n-vector of link delays as the minimizer of the norm squared of the residual of our trip time model. In other words, we choose $\hat{d}$ to minimize $\|Rd - c\|^2$, for some $m \times n$ matrix R and some m-vector c.

Say what R and c are. (That is, give all their entries.)

12.5 Suppose A is a tall matrix with linearly independent columns, and let $\hat{x}$ denote the least squares approximate solution of the over-determined set of equations $Ax = b$. Determine if each of the following statements is true or false. (True means that it must hold, with no other assumptions about A, b, or $\hat{x}$.)

(a) $A\hat{x} = b$.

(b) $A\hat{x} \neq b$.

(c) If $\hat{x} = 0$, then b is orthogonal to the columns of A.

(d) If $A\hat{x} \neq b$, then the system of equations $Ax = b$ does not have a solution.

(e) $\|A\hat{x} - b\| \leq \|b\|$.

12.6 *Estimating a vector from noisy measurements of sums of pairs of entries.* We wish to estimate a 4-vector x, given noisy measurements of all pairs of its entries, *i.e.*, $y_{i,j} \approx x_i + x_j$, for $1 \leq i < j \leq 4$. (The numbers $y_{i,j}$ are given.) You will estimate x by minimizing the sum of squares of the residuals $x_i + x_j - y_{i,j}$ over all such pairs. Express this objective in the form $\|Ax - b\|^2$, for an appropriate matrix A and vector b. You must state the dimensions of A and b, and given their entries.

Hint. There are six pairs of entries.

12.7 *Constructing a portfolio of bonds to approximate a sequence of liabilities.* A business has a (predicted) sequence of monthly liabilities (payments it must make) over 10 years, given by the 120-vector l. The business will purchase 20 different bonds, with quantities given by the 20-vector q. Bond i is associated with a sequence of monthly coupon payments given by the 120-vector c_i.

(There are many different types of bonds, with different coupon payments. Some pay every month; others pay every 6 or 12 months. Each bond has a maturity date, after which the coupon payments are zero. But you don't need to know this.)

Let p denote the 120-vector of total coupon payments from the bonds. It is given by $p = q_1 c_1 + \cdots + q_{20} c_{20}$. The bond quantities will be chosen by minimizing $\|l - p\|^2$, which means we are trying to match the payments to the liabilities. Explain how to set this up as the problem of minimizing $\|Aq - b\|^2$, where A is a matrix and b is a vector. You must say what A and b are, and give their dimensions.

Remark. The liabilities are nonnegative, as are the coupon payments, but we ignore that here.

12.8 *Constant input that minimizes the state norm in N periods.* Consider the linear dynamical system $x_{t+1} = Ax_t + Bu_t$, where the state x_t is an n-vector and the input u_t is an m-vector, with $m < n$. Suppose we apply a constant input w in periods $t = 1, \ldots, N-1$, *i.e.*, $u_1 = u_2 = \cdots = u_{N-1} = w$. What is the value of w that minimizes $\|x_N\|^2$? We are expecting a formula for w that involves A, B, N, and x_1. You can use the pseudo-inverse in your formula. You can assume that a matrix arising in your formula has linearly independent columns.

Hint. First derive an expression for x_N in terms of w and A, B, N, and x_1.

13 Least squares data fitting

13.1 *Saving TA time using midterm score prediction.* The TAs very carefully graded all problems on all students' midterms. But suppose they had just graded the first half of the exam, *i.e.*, problems 1–5, and used a regression model tor predict the total score on each exam.

The matrix of midterm scores is available on the midterm page on the course web site. There rows correspond to students (in random order, anonymized), and the columns to midterm questions.

(a) Find the average and standard deviation of the total midterm scores. Find the average and standard deviation for each individual midterm problem.

(b) Here is a very simple way to predict the total score based on the 5 scores for first half of the exam: Sum the 5 first half scores, and double the result. What RMS prediction error does this simple method achieve?

(c) Find a regression model that predicts the total score based on the 5 scores for problems 1–5. Give the coefficients and briefly interpret them. What is the RMS prediction error of your model?

(Just for fun, evaluate the predictor on your own exam score. Would you rather have your actual score or your predicted score?)

Remark. Your dedicated ENGR108 teaching staff would *never* do anything like this. Really.

13.2 *Moore's law.* These numbers are available in the Julia file `moore_data.jl` In this data file, `t` is the first column of the table (introduction year) and `N` is the second column (number of transistors). *Hint.* In Julia, the function $\log_{10}$ is `log10`.

13.3 *Auto-regressive time series prediction.* Suppose that x is an N-vector representing time series data. The (one step ahead) prediction problem is to guess x_{t+1}, based on $x_1, \ldots, x_t$. We will base our prediction $\hat{x}_{t+1}$ of x_{t+1} on the previous M values, $x_t, x_{t-1}, \ldots, x_{t-M+1}$. (The number M is called the *memory length* of our predictor.) When the prediction is a linear function,

$$\hat{x}_{t+1} = \beta_1 x_t + \beta_2 x_{t-1} + \cdots + \beta_M x_{t-M+1},$$

it is called an *auto-regressive predictor.* (It is possible to add an offset to $\hat{x}_{t+1}$, but we will leave it out for simplicity.) Of course we can only use our auto-regressive predictor for $M \leq t \leq N-1$.

Some very simple and natural predictors have this form. One example is the predictor $\hat{x}_{t+1} = x_t$, which guesses that the next value is the same as the current one. Another one is $\hat{x}_{t+1} = x_t + (x_t - x_{t-1})$, which guesses what x_{t+1} is by extrapolating a line that passes through x_t and x_{t-1}.

We judge a predictor (*i.e.*, the choice of coefficients β_i) by the mean-square prediction error

$$J = \frac{1}{N-M} \sum_{t=M}^{N-1} (\hat{x}_{t+1} - x_{t+1})^2.$$

A sophisticated choice of the coefficients β_i is the one that minimizes J. We will call this the least-squares auto-regressive predictor.

(a) Find the matrix A and the vector b for which $J = \|A\beta - b\|^2/(N-M)$. This allows you to find the coefficients that minimize J, *i.e.*, the auto-regressive predictor that minimizes the mean-square prediction error on the given time series. Be sure to give the dimensions of A and b.

(b) For $M = 2, \ldots, 12$, find the coefficients that minimize the mean-square prediction error on the time series `x_train` given in `time_series_data.jl`. The same file has a second time series `x_test` that you can use to test or validate your predictor on. Give the values of the mean-square error on the train and test series for each value of M. What is a good choice of M? Also find J for the two simple predictors described above.

Hint. Be sure to use the `toeplitz` function contained in `time_series_data.jl`. It'll make your life alot easier. Documentation for the function is also contained in `time_series_data.jl`.

13.4 *Modeling class attendance.* A university registrar keeps track of class attendance, measured as a percentage (so 100 means full attendance, and 30 means 30% of the students attend). For each class lecture, she records the attendance y, and several features:

- x_1 is the day of week, with Monday coded as 1 and Friday coded as 5.
- x_2 is the week of the quarter, coded as 1 for the first week, and 10 for the last week.
- x_3 is the hour of the lecture, with 8AM coded as 8, and 4PM coded as 16.
- $x_4 = \max\{T - 80, 0\}$, where T is the outside temperature (so x_4 is the number of degrees above 80°F).
- $x_5 = \max\{50 - T, 0\}$, where T is the outside temperature (so x_5 is the number of degrees below 50°F).

(These features were suggested by a professor who is an expert in the theory of class attendance.) An ENGR108 alumna carefully fits the data with the following regression model,

$$\hat{y} = -1.4x_1 - 0.3x_2 + 1.1x_3 - 0.6x_4 - 0.5x_5 + 68.2,$$

and validates it properly.

Give a short story/explanation, in English, of this model.

13.5 *Nonlinear auto-regressive model.* We have a (scalar) time series $z_1, z_2, \ldots, z_T$. The following one step ahead prediction model is proposed:

$$\hat{z}_{t+1} = \theta_1 z_t + \theta_2 z_{t-1} + \theta_3 z_t z_{t-1},$$

where $\theta = (\theta_1, \theta_2, \theta_3)$ is the model parameter vector. The sum of the squares of the prediction error of this model on the given time series is

$$\sum_{t=2}^{T-1} (\hat{z}_{t+1} - z_{t+1})^2.$$

(Note that we must start this sum with $t = 2$, since z_0 and z_{-1} are not defined.) Express this quantity as $\|A\theta - b\|^2$, where A is a $(T-2) \times 3$ matrix and b is a $(T-2)$-vector. (You must say what the entries of A and b are. They can involve the known data $z_1, \ldots, z_T$.)

Remark. Finding A and b is the first step in fitting the parameters θ to the data. We are not asking you to find θ, but only to set up the least squares problem you'd solve to carry out the least squares fit.

13.6 *Lecture attendance, laptops in lecture, and final grade.* A study collects data on a large number of students in a lecture course, with the goal of predicting the effect (or at least the association) of lecture attendance and laptop use on the final exam grade. The regressor is the 2-vector x, where x_1 is the student's lecture attendance, expressed as a number between 0 and 1 (with, say, 0.78 meaning the student attended 78% of the lectures), and x_2 is a Boolean feature that codes whether or not the student routinely used a laptop during lecture, with $x_2 = 0$ meaning she did not, and $x_2 = 1$ meaning that she did. The outcome variable y is the student's final exam grade, scaled to be between 0 and 100 points. A basic regression model $\hat{y} = x^T\beta + v$ is fit to the data, and checked with out-of-sample validation.

(a) Give a one sentence interpretation of β_1.

(b) Give a one sentence interpretation of β_2.

(c) Suggest a value of β_1 that would not surprise you, and give a one sentence explanation that might be appropriate if the value of β_1 were your value.

(d) Suggest a value of β_2 that would not surprise you, and give a one sentence explanation that might be appropriate if the value of β_2 were your value.

For parts (c) and (d) we are looking for a plausible guess of the values, along with a plausible story that would go along with your guessed value.

13.7 A daily time series of demand for some product, denoted y_t for day t, has been standardized based on its historical average and standard deviation, so $y_t = 0$ means the demand on day t has its historical average value, and $y_t = -1$ means the demand is one standard deviation below its historical average. When y_t is positive, say bigger than 0.3 or so, we say it is high demand day, and when y_t is negative, say smaller than -0.3 or so, we say it is a low demand day.

A data scientist develops a regression model $\hat{y}_{t+1} = x_t^T\beta + v$ that predicts tomorrow's demand y_{t+1} from the following two features: $(x_t)_1 = y_t$ is today's demand, and $(x_t)_2 = 1$ if tomorrow is a weekday, and $(x_t)_2 = 0$ if tomorrow is a weekend or official holiday. She finds that the model makes reasonably good predictions on test data, with RMS error around 0.1. Her model coefficients are $\beta = (0.8, -0.2)$, $v = 0.1$. Based on her model, which of the following statements are reasonable?

(a) High demand days are typically followed by high demand days.

(b) Low demand days are typically followed by high demand days.

(c) Demand is typically much higher on weekends and holidays, compared to weekdays.

(d) Demand typically decreases a bit on weekdays, compared to weekends and holidays.

13.8 *Fitting a piecewise linear function to data.* We are given some samples (x_i, y_i), $i = 1, \ldots, N$, with x_i and y_i scalars, with a function of the form form

$$\hat{y} = \hat{f}(x) = \theta_1 + \theta_2 \min\{x, 0\} + \theta_3 \max\{x, 0\},$$

where $\theta = (\theta_1, \theta_2, \theta_3)$ contains the parameters. The function $\hat{f}$ is affine for $x \leq 0$, and also for $x \geq 0$, and it is continuous at $x = 0$ with value θ_1. Such a function is called *piecewise affine*, or more commonly, *piecewise linear*, with a single *knot* or *kink point* at 0. We choose the parameters using the least squares fit on the given data points.

Consider the specific data set of (x, y) pairs

$$(-2, 3), \quad (-1, 1), \quad (0, 1), \quad (1, 2), \quad (1, 1).$$

Give the matrix A and vector b for which the sum of squares of the fitting errors is equal to $\|A\theta - b\|^2$. We want the explicit numerical values of A and b. We are not asking you to solve the least squares problem, *i.e.*, find the $\hat{\theta} = A^\dagger b$.

13.9 *Least squares interpolation.* Suppose the N-vector z represents a time series. Interpolation refers to guessing the value z_t from other values of z_τ, for τ both smaller than t (*i.e.*, past values) and also τ larger than t (*i.e.*, the future). (When we guess z_t based on values z_τ for $\tau < t$, we call it prediction.)

We consider an interpolator that uses the two values before and after z_t to guess z_t. It has the form

$$\hat{z}_t = \theta_1 z_{t-2} + \theta_2 z_{t-1} + \theta_3 z_{t+1} + \theta_4 z_{t+2}, \quad t = 3, \ldots, N-2.$$

(There is nothing special about using two values before and after t; this interpolation method is easily extended to using any number of values before, and any number of values after, t.)

We will use least squares to fit the parameters $\theta = (\theta_1, \ldots, \theta_4)$, using the time series z as our training data. That is, we choose θ to minimize the sum square error

$$\sum_{t=3}^{N-2} (\hat{z}_t - z_t)^2.$$

For the specific case $N = 9$, express this as $\|A\theta - b\|^2$, where A is a matrix and b is a vector. Give the size of A and b, and all of their entries (which can involve the entries of the time series z). *Note.* In practice, you'd probably want to fit this model on a time series with a larger value of N.

13.10 *Sign-dependent regression model.* Consider a fitting problem with a single scalar feature x and outcome y. A simple regression model has the form $\hat{y} = \beta x + v$, with model parameters (scalars) β and v. A sign-dependent regression model treats the feature x differently when it is negative and positive:

$$\hat{y} = \beta_1 \min\{x, 0\} + \beta_2 \max\{x, 0\} + v,$$

with parameters β_1, β_2, and v. (You can think of this as simple feature engineering, where we create two new features from the one original feature.)

The simple regression model and the sign-dependent regression model are fit using least squares on the same training data set, and both are evaluated on the same test set. (You can assume that the matrices arising in the two fitting problems have linearly independent columns.)

Which of the following statements are true, *i.e.*, always hold?

(a) The training RMS error of the sign-dependent model is at least as small as the training RMS error of the simple regression model.

(b) The test RMS error of the sign-dependent model is at least as small as the test RMS error of the simple regression model.

Which of the following statements are reasonable? (Meaning, do you agree with the statement?)

(c) The sign-dependent model is more expressive than the simple regression model, and so should always be used.

(d) The sign-dependent model will likely be over-fit, so we should always use the simple regression model.

(e) Validation or cross-validation should be used to determine which of the two models to use.

13.11 *Basis functions with non-overlapping support.* Consider a linear in the parameters prediction model of the form

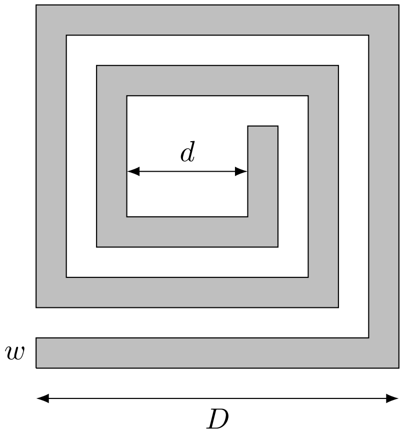
$$\hat{y} = \theta_1 f_1(x) + \cdots + \theta_p f_p(x),$$

where the n-vector x is the feature vector, the scalar $\hat{y}$ is our prediction, $\theta_1, \ldots, \theta_p$ are the model parameters, and $f_i : \mathbf{R}^n \to \mathbf{R}$, $i = 1, \ldots, p$ are the basis functions. The *support* of a function is the set of points where its value is nonzero. We say that two basis functions f_i and f_j, with $i \neq j$, have *non-overlapping support* if there is no x for which both are nonzero, *i.e.*, if $f_i(x) f_j(x) = 0$ for all x. As a simple example with $n = 1$, consider $f_1(x) = \max\{x, 0\}$ and $f_2(x) = \min\{x, 0\}$.

Now consider a set of data with N samples, $x^{(1)}, \ldots, x^{(N)}$, $y^{(1)}, \ldots, y^{(N)}$, and let A denote the associated data matrix, with $A_{ij} = f_j(x^{(i)})$ for $i = 1 \ldots, N$ and $j = 1, \ldots, p$. (See (13.1) on page 247 of the book.) Let $G = A^T A$ be the associated Gram matrix. If basis functions f_i and f_j have non-overlapping support, what can you say about G_{ij}? Be as specific as you can.

13.12 The figure shows a planar spiral inductor, implemented in CMOS, for use in RF circuits. The inductor is characterized by four key parameters:

- n, the number of turns (which is a multiple of 1/4, but that needn't concern us)
- w, the width of the wire
- d, the inner diameter
- D, the outer diameter.



The inductance L of such an inductor is a complicated function of the parameters n, w, d, and D. It can be found by solving Maxwell's equations, which takes considerable computer time, or by fabricating the inductor and measuring the inductance. In this problem you will develop a simple approximate inductance model of the form

$$\hat{L} = \alpha n^{\beta_1} w^{\beta_2} d^{\beta_3} D^{\beta_4},$$

where $\alpha, \beta_1, \beta_2, \beta_3, \beta_4 \in \mathbf{R}$ are constants that characterize the approximate model. (Since L is positive, we have $\alpha > 0$, but the constants $\beta_2, \ldots, \beta_4$ can be negative.) This simple approximate model, if accurate enough, can be used for design of planar spiral inductors.

The file `inductordata.jl` contains data for 50 inductors, obtained from measurements. The Julia command `include("inductordata.jl")` creates 5 vectors n, w, d, D, L of length 50. The ith elements of these vectors are the parameters n_i, w_i (in μm), d_i (in μm), D_i (in μm) and the inductance L_i (in nH) for inductor i. Thus, for example, w_{13} gives the wire width of inductor 13. Your task is to find α, $\beta_1, \ldots, \beta_4$ so that

$$\widehat{L}_i = \alpha n_i^{\beta_1} w_i^{\beta_2} d_i^{\beta_3} D_i^{\beta_4} \approx L_i, \quad i = 1, \ldots, 50.$$

Your solution must include a clear description of how you found your parameters, as well as their actual numerical values. Note that we have not specified the criterion that you use to judge the approximate model (*i.e.*, the fit between $\widehat{L}_i$ and L_i); we leave that to your judgment.

We can define the *percentage error* between $\widehat{L}_i$ and L_i as

$$e_i = 100 \frac{|\widehat{L}_i - L_i|}{L_i}.$$

Find the average percentage error for the 50 inductors, *i.e.*, $(e_1 + \cdots + e_{50})/50$, for your model. (We are only asking you to find the average percentage error for your model; we do not require that your model minimize the average percentage error.)

13.13 *Efficient leave-one-out cross validation.* Leave one out cross validation (LOOCV) involves repeatedly fitting a linear regression model on all *except* a single datapoint, and then evaluating the model on the held out point. Consider data in pairs (a_i, y_i), $i = 1, \ldots, m$, where $y_i \in \mathbf{R}$ and a_i are n-vectors, and assume the Gram matrix $A^T A$ is invertible, and that $0 \leq a_i^T (A^T A)^{-1} a_i < 1$ for each $i = 1, \ldots, m$. Define $\hat{\theta}_{-i}$ to be the ith leave-one-out prediction vector, that is, $\hat{\theta}_{-i}$ minimizes the leave-one-out objective

$$J_i(\theta) := (a_1^T \theta - y_1)^2 + \cdots + (a_{i-1}^T \theta - y_{i-1})^2 + (a_{i+1}^T \theta - y_{i+1})^2 + \cdots + (a_m^T \theta - y_m)^2.$$

(a) Show that $\hat{\theta}_{-i} = (A^T A - a_i a_i^T)^{-1} (A^T y - a_i y_i)$.

We will be able to efficiently compute the leave-one-out predictions

$$\hat{y}_{-i} = a_i^T \hat{\theta}_{-i}$$

using the Woodbury matrix identity in Question 11.27. To do so, define the $m \times m$ matrix H to have entries $H_{ij} = a_i^T (A^T A)^{-1} a_j$ for $i, j = 1, \ldots, m$.

(b) Show that if $\hat{\theta} = (A^T A)^{-1} A^T y$ is the set of linear regression coefficients using the full data, then

$$\hat{y}_{-i} = a_i^T \hat{\theta}_{-i} = \frac{a_i^T \hat{\theta} - H_{ii} y_i}{1 - H_{ii}}.$$

(c) Show that the mean leave-one-out error, meaning the sum of squared errors of the predictions $\hat{y}_{-i}$ for y_i, is

$$\frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_{-i})^2 = \frac{1}{m}\sum_{i=1}^{m}\frac{(a_i^T\hat{\theta} - y_i)^2}{(1 - H_{ii})^2}.$$

(d) Explain how to compute the LOOCV error in order mn^2 flops.

14 Least squares classification

14.1 *Trading off false positive and false negative rates.* In this problem you use the Boolean least squares classifier with skewed decision point, described on pages 223-224 of the textbook, to alter the false positive and false negative rates. The classifier is $\hat{f}(x) = \mathbf{sign}(\tilde{f}(x) - \alpha)$, where $\tilde{f}$ is the (real-valued) least squares predictor of the labels encoded as ±1, and α is a constant that skews the decision point. We will use basic regression, *i.e.*, $\tilde{f}(x) = x^T\beta + v$.

(a) *Basic classifier.* The file `classifier_with_tradeoff_data.jl` contains train and test sets `X_train`, `X_test`, and the corresponding labels `y_train` and `y_test`. Find the model parameters β and v using the training data set, and give the confusion matrix for the associated classifier (with $\alpha = 0$) for both the training and test sets. Give the error rate for the train and test sets.

(b) *ROC curve from skewed decision point classifier.* The ROC curve plots the performance of different classifiers with the vertical axis showing the true positive rate, and the horizontal axis showing the false positive rate. (See textbook, page 226.) Give ROC plots for the training and test data sets, for the skewed decision point classifier, for 20 values of α ranging from -0.3 to $+0.3$. (You will use the values of β and v found in part (a).)

Julia hints.

- If `y_til` is the vector $\tilde{y}$ (the continuous regression prediction), you can find $\mathbf{sign}(\tilde{y} - \alpha)$ using `sign(y_til-alpha)`.
- To find the number of true positives, you can use `sum((y_true .== 1) & (y_hat .== 1))`. (Similar constructions give the number of false positives, false negatives, and so on.)

14.2 *Equalizer design from training message.* (Continues book exercise 14.10.) The file `eq_design_data.jl` contains the training message s^{tr}, the value of n, and the signal received from the training message, y^{tr}. Your first task is to design an equalizer h using this data, and plot it.

The file also includes the received signal y from a message s that is sent. First, take the sign of y to get an estimate of the message, and print it as a string. Then estimate the message s using your equalizer, and print it as text. You'll know when your equalizer is working.

Hints.

- In Julia you can take the sign of a vector using `sign.(x)`.
- You can turn a Boolean vector with entries encoded as -1 and 1 into a string using the function `binary2string`. (While not needed for this problem, the function `string2binary` converts a string to a Boolean vector.)

14.3 *Iris classification.* In this exercise you will develop and test a 3-class classifier for classifying the flower Iris into three species: *iris setosa*, *iris versicolor*, and *iris virginica*, which we will refer to as classes 1, 2, and 3, respectively. The classifier will use $n = 4$ features:

$$x = (\text{sepal length},\ \text{sepal width},\ \text{petal length},\ \text{petal width})\,.$$

(You don't need to know what these are.) The data set, which contains 150 examples, is a classic one, first published in 1936 by the famous statistician Ronald Fisher. (At the very least, you might think about how much fun it would be to carry out the calculations needed in this exercise in 1936, by hand.)

You will find the data set in `iris_flower_data.jl`. You can use the first 100 examples (2/3) for training, and the last 1/3 for testing. (We have randomized the order of the examples in the data set given.)

(a) Find 3 least squares regression classifiers, each one classifying one of the species against the other two. Give the error rate for each classifier, on both the train and test sets.

(b) Combine the three classifiers of part (a) into a 3-class classifier, and give the 3×3 confusion matrix for the train and test sets.

Hints. The Julia code snippet `2(y .== k) - 1` might be useful. This checks if each entry of vector `y` is equal to `k`. If it is, then the result for that entry is `1`; otherwise it is `-1`. Additionally, it may be useful to consult our helper file, `iris_multiclass_helpers.jl`. We have provided you two functions. The first is `argmax_by_row(A)`, which computes and returns a column vector x, where x_i is the index of the maximum entry in the ith row of `A`. The second is `confusion_matrix(y_hat, y_true)`, which computes and returns a confusion matrix C. For example, C_{11} is the number of predictions where your classifier correctly predicted that the flowers belong to class 1, *iris setosa*.

14.4 *Modifying a classifier.* A co-worker develops a classifier of the form $\hat{y} = \mathbf{sign}(x^T\beta + v)$, with $v < 0$, where the n-vector x is the feature vector, and the n-vector β and scalar v are the classifier parameters. The classifier is evaluated on a given test data set. The false positive rate is the fraction of the test data points with $y = -1$ for which $\hat{y} = 1$. (We will assume there is at least one data point with $y = -1$.)

Are each of the following statements true or false? True means it always holds, with no other assumptions on the data set or model; false means that it need not hold.

(a) Replacing v with zero will reduce, or not increase, the false positive rate.

(b) Replacing β with zero will reduce, or not increase, the false positive rate.

(c) Halving v (*i.e.*, replacing v with $v/2$) will reduce, or at least not increase, the false positive rate.

(d) Halving β (*i.e.*, replacing β with $(1/2)\beta$) will reduce, or at least not increase, the false positive rate.

14.5 A least squares classifier $\hat{y} = \mathbf{sign}(x^T\beta + v)$ achieves false positive rate 0.07 and false negative rate 0.15 on the training set, and false positive rate 0.08 and false negative rate 0.14 on a test set. To reduce the false negative rate, how should we adjust the offset v? Increase it or decrease it?

14.6 *Confusion matrix from error rates.* A Boolean classifier is tested on a set of data with a total number of data points N, broken down into the confusion matrix entries: N_{tp} true positives, N_{fn} false negatives, N_{fp} false positives, and N_{tn} true negatives, with $N_{\text{tp}} + N_{\text{fn}} + N_{\text{fp}} + N_{\text{tn}} = N$.

The classifier achieves an error rate E, a true positive rate T, and a false positive rate F, defined as

$$E = (N_{\text{fp}} + N_{\text{fn}})/N, \quad T = N_{\text{tp}}/(N_{\text{tp}} + N_{\text{fn}}), \quad F = N_{\text{fp}}/(N_{\text{fp}} + N_{\text{tn}}).$$

Can we recover the confusion matrix entries N_{tp}, N_{fn}, N_{fp}, and N_{tn} from N, E, T, and F? The answer is yes.

Express the relations given above as a square set of linear equations in the 4-vector $x = (N_{\text{tp}}, N_{\text{fn}}, N_{\text{fp}}, N_{\text{tn}})$, in the form $Ax = b$. Give A and b. Their entries can depend on N, E, T, and F. We will take points off for answers that are correct but more complicated than they need to be.

Note. We are not asking you to analytically solve this set of equations, or to justify that A is invertible.

14.7 *Efficient leave-one-out cross validation.* In this question, you will implement the efficient leave-one-out cross validation (LOOCV) methods you discovered in Question 13.13. Using the data loading file `loocv_starter.jl` (or the alternative file `loocv_csv_starter.jl`), use the provided methods to load data $m \times n$ data matrices `Atrain`, `Atest`, and label vectors `ytrain` and `ytest`, where the data matrices of size $m \times n$ have entries in $[0, 1]$ and `ytrain` and `ytest` contain entries in $\{0, 1, \ldots, 9\}$ corresponding to the 10 handwritten digits.

Implement code to do the following:

i. Convert the training and test label vectors into vectors y^{train} and y^{test}, whose sizes match those of `ytrain` and `ytest`, where $y_i^{\text{train}} = +1$ if the ith training label is a 0 and $y_i^{\text{train}} = -1$ if the ith training label is not a 0. Do the same for y^{test} and `ytest`.

Let $a \in \mathbf{R}^n$ stand for the (raw) pixel features of a single example. For a given dimension $d \in \{0, 1, 2, \ldots\}$, consider the following random feature generation: generate a random matrix $W \in \mathbf{R}^{n \times d}$, where the ith column w_i of W has entries each chosen uniformly at random in $\{-1/\sqrt{n}, 1/\sqrt{n}\}$. Then generate the *featurization* of a by

$$\phi(a) = (a_1, \ldots, a_n, \cos(w_1^T a), \cos(w_2^T a), \ldots, \cos(w_d^T a)) \in \mathbf{R}^{n+d}.$$

ii. For each dimension $d = 0, 200, 400, 800, 1600$, generate a training and test matrix X^{train} and X^{test} by applying the featurization ϕ to each row of `Atrain` and `Atest`. For a given dimension d, use the *same* random matrix $W \in \mathbf{R}^{n \times d}$ to do this to both the training and test data. *Hint.* The `julia` method `hcat` will concatenate two matrices together horizontally. So, for example,

```
hcat(Atrain, cos.(Atrain * W))
```

will construct a matrix of size $m \times (n + d)$, assuming `W` has dimensions $n \times d$ and `Atrain` has dimensions $m \times n$.

iii. Use the efficient LOOCV approach in Question 13.13 to perform leave-one-out cross validation for the misclassification error rate of a fit model. *Hint.* A predicted value $\hat{y} = \hat{\theta}^T \phi(a)$ is correct for a label $y \in \{-1, 1\}$ if $\hat{y}y > 0$, and incorrect otherwise. Part (b) of Question 13.13 will prove useful.

Implement the above code, and plot your training error, leave-one-out cross validation error, and test error for random feature dimensions $d = 0, 200, 400, 800, 1600$ (and, if you wish, $d = 3200$). Note that you always should include the raw pixel values as features in your linear model. Include your code and plots in your solution. *Remark.* On a 2021 Apple MacBook Pro (32GB RAM and an M1 processor), it took our code about 15 seconds to perform leave-one-out CV with 0, 200, 400, 800, and 1600 random features (combined).

15 Multi-objective least squares

15.1 *Trading off tracking error and input size in control.* A system that we want to control has input (time series) n-vector u and output (time series) y, related by convolution: $y = h * u$, where h is an m-vector. (So y is an $(n+m-1)$-vector.) (See §15.2 in the textbook.) We are given y^{des}, the (time series of) desired or target output values, and we will choose the input u to minimize $\|y - y^{\text{des}}\|^2 + \lambda\|u\|^2$, where $\lambda > 0$ is a parameter we use to trade off tracking error (*i.e.*, $\|y - y^{\text{des}}\|^2$) and input size (*i.e.*, $\|u\|^2$).

We will consider the case where $n = 100$, $m = 7$, with

$$h = (0.3, 0.5, 0.6, 0.4, 0.3, 0.2, 0.1).$$

The desired output is the 106-vector

$$y_t^{\text{des}} = \begin{cases} 10 & 10 \le t < 40 \\ -5 & 40 \le t < 80 \\ 0 & \text{otherwise.} \end{cases}$$

(a) Plot a trade off curve of the RMS tracking error ($\textbf{rms}(y - y^{\text{des}})$) versus the input RMS value ($\textbf{rms}(u)$). As usual with trade-off curves, you should vary λ over a wide range, using values that are logarithmically spaced (*i.e.*, by a constant factor).

(b) Pick 3 values of λ that correspond to too little regularization, too much regularization, and a reasonable amount of regularization. (Reasonable might correspond to RMS tracking error around 0.3) Plot the input u found for each choice of λ on the same figure. Also plot the output y found for each λ on the same figure, along with y^{des}.

Julia hints.

- You will find the following Julia function `logspace` useful for creating a set of values of λ.

  ```
  logspace(start, stop, length=10)=10.^(range(start, stop=stop, length=length))
  ```

- You can use the function `toeplitz` in the VMLS package to express the convolution $y = h * u$ as a matrix vector multiplication, $y = Au$ for some matrix A.

15.2 *Least squares classification with regularization.* You will start by generating synthetic (simulated) data that contains feature n-vectors $x_1, \ldots, x_N$, and the associated binary labels, $y_1, \ldots, y_N$, each of which is either $+1$ or -1. The feature vectors are stored as an $n \times N$ matrix X with columns $x_1, \ldots, x_N$, and the labels are stored as an N-vector y. You'll create both a training and test data set using the code below.

```
Random.seed!(1)
n = 50;
N = 300;
N_test = 100

# Generate training data X, y
b_true = randn(n);
```

```
v_true = 5;
X = randn(n, N)
y = sign.(X'*b_true + v_true + 10*randn(N))

# Generate test data X_test y_test
X_test = randn(n, N_test)
y_test = sign.(X_test'*b_true + v_true + 10*randn(N_test))
```

(a) *Least squares classifier.* Find β, v that minimize $\sum_{i=1}^{N}(x_i^T\beta + v - y_i)^2$ on the training set. Our predictions are then $\hat{f}(x) = \mathbf{sign}(x^T\beta + v)$. Report the classification error on the training and test sets, the fraction of examples where $\hat{f}(x_i) \neq y_i$. There is no need to report the β, v values.

(b) *Regularized least squares classifier.* Now we add regularization to improve the generalization ability of the classifier. Find β, v that minimize

$$\sum_{i=1}^{N}(x_i^T\beta + v - y_i)^2 + \lambda\|\beta\|^2,$$

where $\lambda > 0$ is the regularization parameter, for a range of values of λ. We suggest the range 10^{-1} to 10^4, say, 100 values logarithmically spaced. Plot the training and test set errors versus λ with a logarithmic horizontal axis. Suggest a reasonable choice of λ. Again, there is no need to report the β, v values; just attach the plot and give a reasonable value of λ.

Julia hints.

- You might create a function `error_rate(y,yhat)` such as

  ```
  error_rate(y,yhat) = sum(y.!=yhat)
  ```

- You will find the following function useful for generating an array of values that are logarithmically spaced between two values:

  ```
  logspace(start, stop, length=10)=10.^(range(start, stop=stop, length=length))
  ```

- To create a plot with a logarithmic horizontal scale, use something like

  ```
  plot(lambdas, errors, xaxis=:log)
  ```

15.3 *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 is 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 units of the prices and demands don't really matter in this problem. Demand could be measured in 10000 units, and prices in \$100.

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. You might find it easier to separately fit the models $\delta_i^d \approx \tilde{e}_i^T \delta^p$, where $\tilde{e}_i^T$ is the ith row of E. (We use the tilde above e_i to avoid conflict with the notation for unit vectors.)

Carry out your method using the price and demand data in the matrices `Prices` and `Demands`, found in `price_elasticity.jl`. Give your estimate $\hat{E}$, and guess (roughly) how accurate your model $\delta^d = \hat{E}\delta^p$ might be (in terms of RMS prediction error) on unseen data.

Here are some facts about elasticity matrices that might help you check that your estimates make sense (but you don't need to incorporate this information into your estimation method). The diagonal entries of E are always negative, and typically on the order of one. (This means that when you raise the price of one product only, demand for it goes down by a similar fractional amount as the price increase.) The off-diagonal entries can have either sign, and are typically (but not always) smaller than one in magnitude.

15.4 *Regularized least squares in Julia.* You are asked to write some Julia code to compute the $\hat{x}$ that minimizes $\|Ax - b\|^2 + \lambda\|x\|^2$, where A is an $m \times n$ matrix, b is an m-vector, and λ is a positive scalar. These are given as the Julia quantities `A`, `b`, and `lambda`, respectively, and the dimensions m and n are given as `m` and `n`. You are to put the value of $\hat{x}$ in `xhat`.

Which of the following Julia snippets will carry this out correctly? Circle the correct response for each code snippet below. You do not need to justify your responses. (You are welcome to try to run the snippets in Julia, but you should make your guesses before doing this.)

(a) `xhat = [A sqrt(lambda)*eye(n)]\b` *Works.* *Doesn't work.*

(b) `xhat = [A sqrt(lambda)*eye(n)]\[b; zeros(m)]` *Works.* *Doesn't work.*

(c) `xhat = inv(A'*A+lambda*eye(n))*(A'*b)` *Works.* *Doesn't work.*

(d) `xhat = [A; sqrt(lambda)*eye(n)]\b` *Works.* *Doesn't work.*

(e) `xhat = [A; sqrt(lambda)*eye(n)]\[b; zeros(n)]` *Works.* *Doesn't work.*

15.5 *Fitting models to two different but similar populations.* We wish to fit two different models to data from two different but similar populations, for example males and females. The models are given by $\hat{y} = x^T\beta$ for the first group and $\hat{y} = x^T\tilde{\beta}$ for the second group, where x is the n-vector of features, $\hat{y}$ is the prediction, β is the n-vector of model parameters for the first group and $\tilde{\beta}$ is the n-vector of model parameters for the second group. (We can include an offset in the two models by including a feature that is always one.)

Our training data consists of $x^{(1)}, \ldots, x^{(N)}$ and $y^{(1)}, \ldots, y^{(N)}$ from the first population, and $\tilde{x}^{(1)}, \ldots, \tilde{x}^{(N)}$ and $\tilde{y}^{(1)}, \ldots, \tilde{y}^{(N)}$ from the second group. (For simplicity we assume that we have an equal number of training data points in the two groups.)

Our main goal in choosing the parameter n-vectors β and $\tilde{\beta}$ is that the sum of squares of the prediction errors for the first group (using the first model) and the sum of squares of the prediction errors for the second group (using the second model) is small. Our secondary objective is that the two parameter vectors β and $\tilde{\beta}$ are not too different. (This desire is based on our idea that the two groups are similar, so the associated models should not be too different.)

Capture the goals expressed above as a bi-objective least squares problem with variable $\theta = (\beta, \tilde{\beta})$. Identify the primary objective $J_1 = \|A_1\theta - b_1\|^2$ and the secondary objective $J_2 = \|A_2\theta - b_2\|^2$. Give A_1, A_2, b_1 and b_2 explicitly. Your solution can involve the $n \times N$ data matrices X and $\tilde{X}$, whose columns are $x^{(i)}$ and $\tilde{x}^{(i)}$, respectively, and the two N-vectors y^{d} and $\tilde{y}^{\mathrm{d}}$, whose entries are $y^{(i)}$ and $\tilde{y}^{(i)}$, respectively.

15.6 *Approximate solution of linear equations with multiple righthand sides.* Suppose A is a square invertible matrix. We seek x for which $Ax \approx b_i$, $i = 1, \ldots, k$. In other words, we wish to find a value of x that approximately solves the systems of equations $Ax = b_1, \ldots, Ax = b_k$. Two colleagues offer suggestions about how to accomplish this.

- Your colleague Bob proposes a multi-objective least squares approach: choose x to minimize
$$\|Ax - b_1\|^2 + \cdots + \|Ax - b_k\|^2.$$
- Alice, another colleague, proposes the following method: first average the righthand sides to get $\bar{b} = (1/k)(b_1 + \cdots + b_k)$, and then choose x by solving one system of linear equations $Ax = \bar{b}$.

Oscar, a third colleague, says that the two approaches sound different but always end up with the same choice of x. Is Oscar right? Justify your answer, *i.e.*, either explain why the two approaches always give the same x, or give a specific counter-example where they do not.

15.7 Suppose that $\hat{x}$ minimizes $\|Ax - b\|^2 + \lambda\|x - c\|^2$, where $\lambda > 0$. Here the matrix A and the vectors b and c are given, and have dimensions for which the expression above makes sense. Which of the following are true?

(a) $\|A\hat{x} - b\| \leq \|Ac - b\|$.

(b) If $\tilde{x}$ minimizes $\|Ax - b\|^2 + \tilde{\lambda}\|x - c\|^2$, where $\tilde{\lambda} > \lambda$, then $\|\tilde{x} - c\| \leq \|\hat{x} - c\|$.

15.8 Regression models are fit on a training data set using regularized least squares, with five different values of λ, which scales the regularization. The RMS error on the training and test sets are shown below. Which value of λ would you choose?

λ	Train RMS error	Test RMS error
0.1	0.6	2.5
0.3	0.7	2.1
1.0	0.9	1.8
3.0	1.1	1.4
10.0	1.5	1.9

15.9 *Least squares smoothing.* You are given a time series represented by the N-vector y. We seek $\hat{y}$, an approximation of y that is also smooth. (We say that $\hat{y}$ is obtained from y by smoothing; in Electrical Engineering dialect, you would say that $\hat{y}$ is the result of processing y through a low-pass filter.) We do this by choosing $\hat{y}$ to minimize

$$\|\hat{y} - y\|^2 + \lambda\|D\hat{y}\|^2,$$

where D is the $(N-1) \times N$ first difference matrix, with $D_{ij} = 1$ for $j = i+1$, $D_{ij} = -1$ for $j = i$, and $D_{ij} = 0$ for all other i and j, and $\lambda > 0$ is a parameter that trades off how close $\hat{y}$ is to y, and how smooth it is.

Give a formula for $\hat{y}$ in terms of y, D, and λ. Your formula should use standard matrices like 0 or I, and operations like multiplication, inverse, or transpose. We will give partial credit for a solution that uses matrix and vector stacking or slicing; the solution we are looking for is a simple formula that does not use stacking or slicing.

If your formula involves an inverse, be sure to explain why the matrix being inverted is invertible.

15.10 *Regularization in least squares classification.* Consider a least squares classifier with regularization, with regularization parameter λ. We fit the classifier on a training data set, and validate it on a test data set. For each of these two data sets, we evaluate two different metrics: The RMS error, *i.e.*, the RMS error of $\tilde{f}(x) - y$, and the error rate, which is the fraction of points for which $\hat{f}(x) \neq y$. Note that $\tilde{f}(x)$ is the real-valued approximation of y, and $\hat{f}(x) = \mathbf{sign}(\tilde{f}(x))$ is our Boolean-valued prediction.

Suppose we increase λ. Which of the following statements is always true? You do not need to give any justification.

(a) The training RMS error will stay the same or increase.

(b) The test RMS error will stay the same or increase.

(c) The training error rate will stay the same or increase.

(d) The test error rate will stay the same or increase.

15.11 *Matrix pricing of bonds.* A *bond* is a financial instrument that entitles the holder to a (nonnegative) cash flow over some limited period of time, up to its maturity date. We consider N bonds, with cash flows given by the T-vectors $c_1, \ldots, c_N$. (We assume that the bonds all have maturity dates that are less than or equal to T. We have $(c_i)_t = 0$ if the maturity date of bond i occurs before period t.) We assume there is no uncertainty in the cash flows. This is a reasonable assumption for government issued bonds.

A typical bond cash flow is a periodic payment (called the coupon payment) until its maturity, when the payment is the coupon plus the principal value of the bond. But there are other different cash flows for bonds, such as a no-coupon bond with no coupon payments and one payment at maturity, and these details won't matter for this problem.

The price p_i of bond i is modeled as given by its discounted present value, *i.e.*, $p_i \approx P^T c_i$, where the T-vector P is the discount vector. The number P_t gives the present value of \$1 in period t. (A better symbol for P might be d, but P is the traditional symbol.) A simple and naïve model for a discount curve is a simple per-period interest rate r, for which $P_t = (1+r)^{-t}$, $t = 1, \ldots, T$. More sophisticated models of the discount curve allow it to have any value, which roughly speaking means that the interest rate can change depending on the term, *i.e.*, how far in future you're looking. This sometimes referred to as the *term structure of interest rates.*

We cannot directly observe the discount curve P, but we can observe the market prices of the bonds, *i.e.*, $p_1, \ldots, p_M$. A critical task is to estimate or guess the discount curve, based on these known market prices. (Once we guess the discount curve, we can use it to predict the price of another bond we haven't enountered before, from its cash flow.)

Matrix pricing is a standard method that is used to do this. It estimates the discount curve as the minimizer of

$$\sum_{i=1}^{M} (p_i - P^T c_i)^2 + \lambda \|DP\|^2,$$

where D is the $(T-1) \times T$ first difference matrix, and λ is a positive parameter than trades off fitting the observed bond market prices with the smoothness of the discount curve. It is called matrix pricing because, well, there is a matrix involved.

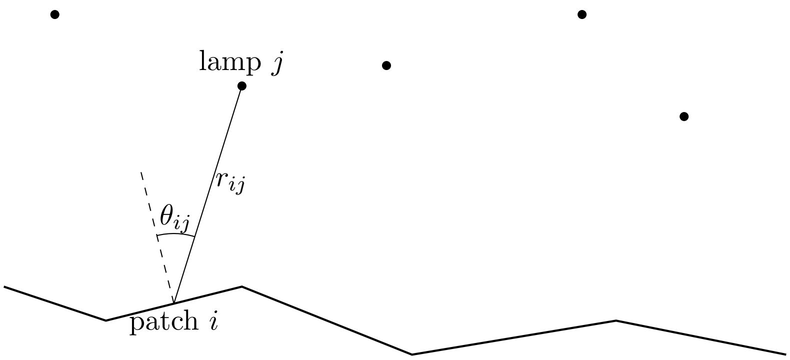
(a) Give a formula for the estimated discount curve $\hat{P}$ using matrix and vector notation. Your formula can include λ, the bond cash flow vectors c_i, the bond market price vector p, and any matrices or vectors that you construct from these. You can use any standard matrix operations, such as addition, multiplication, inverse, pseudo-inverse. You can assume that a matrix occuring in your formula has linearly independent columns.

(b) Suggest a reasonable way to choose the parameter λ. *Hint.* Hold out some bonds as a test set. You can describe your approach in English.

Remarks.

- People in finance usually work with the *yield curve* Y, and not the discount curve P. They are related as $Y_t = P_t^{-1/t} - 1$, and $P_t = (1+Y_t)^{-t}$. The yield Y_t is the constant interest rate over periods $1, \ldots, t$ that would yield the discount P_t in period t.
- The method above is not exactly how yield curves are fit, but it is very similar in spirit.

15.12 The figure shows an illumination system of n lamps illuminating m flat patches. The variables in the problem are the lamp powers $x_1, \ldots, x_n$, which can vary between 0 and 1.



The illumination intensity at (the midpoint of) patch i is denoted I_i. We use a simple linear model for the illumination intensities I_i as a function of the lamp powers x_j: for $i = 1, \ldots, m$,

$$I_i = \sum_{j=1}^{n} A_{ij} x_j.$$

The matrix A (with coefficients A_{ij}) is available in the data file `illum_data.jl` and was constructed as follows. We take

$$A_{ij} = r_{ij}^{-2} \max\{\cos\theta_{ij}, 0\},$$

where r_{ij} denotes the distance between lamp j and the midpoint of patch i, and θ_{ij} denotes the angle between the upward normal of patch i and the vector from the midpoint of patch i to lamp j, as shown in the figure. This model takes into account "self-shading" (*i.e.*, the fact that a patch is illuminated only by lamps in the halfspace it faces) but not shading of one patch caused by another. Of course we could use a more complex illumination model, including shading and even reflections. This just changes the matrix relating the lamp powers to the patch illumination levels.

The problem is to determine lamp powers that make the illumination levels I_i close to a given desired level I_{des}. In other words, we want to choose the n-vector x such that

$$\sum_{j=1}^{n} A_{ij} x_j \approx I_{\text{des}}, \quad i = 1, \ldots, m,$$

but we also have to observe the power limits $0 \leq x_j \leq 1$. This is an example of a *constrained optimization problem.* The objective is to achieve an illumination level that is as uniform as possible; the constraint is that the elements of x must satisfy $0 \leq x_j \leq 1$. Finding the exact solution of this minimization problem requires specialized numerical techniques for constrained optimization. However, we can solve it *approximately* using least squares.

In this problem we consider two approximate methods that are based on least squares, and compare them for the data generated using `include("illumdata.jl")`. This creates an 11×7 matrix A and scalar $I_{\text{des}} = 2$.

(a) *Saturate the least squares solution.* The first method is simply to ignore the bounds on the lamp powers. We solve the least squares problem

$$\text{minimize} \quad \sum_{i=1}^{m} (\sum_{j=1}^{n} A_{ij} x_j - I_{\text{des}})^2$$

ignoring the constraints $0 \leq x_j \leq 1$. If we are lucky, the solution will satisfy the bounds $0 \leq x_j \leq 1$, for $j = 1, \ldots, n$. If not, we replace x_j with zero if $x_j < 0$ and with one if $x_j > 1$.

Apply this method to the problem data generated by `illumdata.jl`, and calculate the resulting value of the cost function $\sum_{i=1}^{m}(I_i - I_{\mathrm{des}})^2$.

(b) *Weighted least squares.* The second method is to solve the problem

$$\text{minimize} \quad \sum_{i=1}^{m}(\sum_{j=1}^{n} A_{ij}x_j - I_{\mathrm{des}})^2 + \lambda \sum_{j=1}^{n}(x_j - 0.5)^2,$$

where the constant $\lambda \geq 0$ is used to attach a cost to the deviation of the powers from the value 0.5, which lies in the middle of the power limits. For $\lambda = 0$, this is the same least squares problem as in part (a). If we take λ large enough, the solution of this problem will satisfy $0 \leq x_j \leq 1$.

Formulate this problem as a least squares problem in the variables x, and solve it for $\lambda = 1$, $\lambda = 2$, $\lambda = 3$, etc., until you find a value of λ such that all components of the solution x satisfy $0 \leq x_j \leq 1$. For that solution x, calculate the cost function $\sum_{i=1}^{m}(I_i - I_{\mathrm{des}})^2$ and compare with the value you obtained in part (a).

16 Constrained least squares

16.1 *Computing least-norm solutions.* Generate a random 10×100 matrix A and a 10-vector b. Use Julia to compute the least norm solution for $Ax = b$ using the methods listed below, and verify that the solutions found are the same (or more accurately, very close to each other). Be sure to submit your code, including the code that checks if the solutions are close to each other.

- Using the formula $\hat{x} = A^T(AA^T)^{-1}b$.
- Using the pseudo inverse: $\hat{x} = A^\dagger b$.
- Using the Julia backslash operator.
- Using the Julia package `LinearLeastSquares`.

16.2 *Julia timing test for least-norm.* Determine how long it takes for your computer to solve a least-norm problem with $m = 600$ equations and $n = 4000$ variables. (You can use the backslash operator.) What approximate flop rate does your result suggest?

Remark. Julia compiles just in time, so you should run the code a few times to get the correct time.

16.3 *Constrained least squares in Julia.* You are asked to write some Julia code to compute the $\hat{x}$ that minimizes $\|Ax - b\|^2$ subject to $Cx = d$, where A is an $m \times n$ matrix, b is an m-vector, C is a $p \times n$ matrix, and d is a p-vector. These are given as the Julia quantities `A`, `b`, `C`, and `d`, and the dimensions m, n, and p are given as `m`, `n`, and `p`. You are to put the value of $\hat{x}$ in `xhat`. (You can assume that the associated KKT matrix is invertible.)

Write two lines of Julia code below that carries this out. Your code should be simple and clear. You do not need to justify your answer.

Hint. Recall that the optimality conditions for this constrained least squares problem are

$$2A^TAx + C^Tz = 2A^Tb, \qquad Cx = d,$$

where z is the vector of Lagrange multipliers.

16.4 *Varying the righthand sides in linearly constrained least squares.* Suppose $\hat{x}$ minimizes $\|Ax - b\|^2$ subject to $Cx = d$. (You can assume that the associated KKT matrix is invertible.) We can think of $\hat{x}$ as a function of b and d (which are called the righthand sides of the linearly constrained least squares problem). A colleague asserts that $\hat{x}$ is a linear function of b and d, and so has the form $\hat{x} = Fb + Gd$ for some appropriate matrices F and G.

Is she right? If not, give a counter-example. If so, justify your answer and explain how to find the matrices F and G.

16.5 *Smallest fund holdings consistent with portfolio return history.* A fund consists of a portfolio with n assets, characterized by the weight n-vector w, which satisfies $\mathbf{1}^Tw = 1$. (Negative entries correspond to short positions.) The fund manager publishes the return of the portfolio on T days, given by $r = Rw$, where R is the $T \times n$ matrix of asset returns, but does not divulge her holdings w. (We do, however, know R.) We consider the case where T is smaller than n, *i.e.*, we are given fewer portfolio returns than there are assets. We will guess w by finding $\hat{w}$, the smallest weight vector (measured by norm squared) that is consistent with the reported portfolio returns (and satisfies $\mathbf{1}^T w = 1$).

Give a formula for $\hat{w}$ (in terms of R, r, and the dimensions n and T). Your formula can involve the usual matrix and vector operations, including matrix addition, transpose, multiplication, inverse, pseudo-inverse, and so on. (If you use inverse or pseudo-inverse, you can assume that the appropriate condition, such as linear independence of columns or rows, holds.)

16.6 *Minimum jerk trajectory.* The N-vector p denotes a time series of positions of an object such as a vehicle. (For simplicity, we consider here the case when it moves in one dimension.) The velocity v, acceleration a, and jerk (third derivative) j are defined as

$$v = D_N p, \qquad a = D_{N-1} v, \qquad j = D_{N-2} a,$$

where D_k denotes the $(k-1) \times k$ first difference matrix. (These are $N-1$, $N-2$, and $N-3$ vectors, respectively.)

(a) Express j in the form $j = Fp$, where F is a Toeplitz matrix. Give the dimensions of F, and explain what its first row is. Give F explicitly for the case $N = 6$; you can guess the general form from that.

(b) Suppose we require

$$p_1 = p^{\text{init}}, \quad p_N = p^{\text{term}}, \quad v_1 = v^{\text{init}}, \quad v_N = v^{\text{term}},$$

where p^{init}, p^{term}, v^{init}, and v^{term} are given initial position and velocity and terminal position and velocity. In addition to satisfying these constraints, we wish to minimize the sum square jerk, $\|j\|^2$.

Express this as a linearly constrained least squares problem with variable p, *i.e.*, minimize $\|Ap - b\|^2$ subject to $Cp = d$. Say what A, b, C, and d are, and give their dimensions. You can refer to the matrix F found in part (a), and also the given initial and terminal positions and velocities.

We are only asking you to set the problem up as a linearly constrained least squares problem. We are not asking you to solve it.

16.7 *De-biasing via linearly constrained least squares.* Prediction models are fit to historical data. When the historical data contains biases, these biases are captured in the model. If the model is then used to make future decisions, it can help to perpetuate the biases. As a specific example consider the problem of predicting interviewer ratings from attributes of a candidate that are culled from their resume. The model is fit using historical data, where due to conscious and unconscious bias, interviewers have given lower ratings to women on average. If the prediction model captures these biases, and is then used to screen applicants (which is a typical use), it will help perpepuate biases.

There is a whole field devoted to various aspects of this problem, such as detecting bias in data and models, and fitting data in ways that reduce the problem of perpetuating bias. In this exercise we examine a simple method based on ideas from the course.

We consider the specific case mentioned above. The historical (training) data consists of pairs $(x^{(i)}, y^{(i)})$, $i = 1, \ldots, N$, where $x^{(i)}$ is an n-vector of features culled from a candidate's resume, and $y^{(i)}$ is the numerical rating given by the interviewer. We assume that the first N^{f} of the candidates are female and the remaining N^{m} candidates are male. (We have $N^{\mathrm{f}} + N^{\mathrm{m}} = N$, and we assume N^{f} and N^{m} are both positive.) We will fit a regression model to this data, *i.e.*, $\hat{y} = x^T\beta + v$, where the n-vector β and number v are the model parameters. We choose β and v by minimizing the sum square prediction error on the training data,

$$\sum_{i=1}^{N} (\hat{y}^{(i)} - y^{(i)})^2,$$

where $\hat{y}^{(i)} = (x^{(i)})^T\beta + v$. We also impose an *anti-bias constraint*, which is that the average predicted score for the female and male candidates on the training set are equal, *i.e.*,

$$\frac{1}{N^{\mathrm{f}}} \sum_{i=1}^{N^{\mathrm{f}}} \hat{y}^{(i)} = \frac{1}{N^{\mathrm{m}}} \sum_{i=N^{\mathrm{f}}+1}^{N} \hat{y}^{(i)}.$$

Explain how to formulate this as a linearly constrained least squares problem with variable (β, v) of the form

$$\begin{array}{ll} \text{minimize} & \|A(\beta, v) - b\|^2 \\ \text{subject to} & C(\beta, v) = d. \end{array}$$

Give A, b, C, and d, making sure to specify their dimensions and entries. You can use X, the $n \times N$ matrix of features, with columns $x^{(i)}$, and the N-vector y^{d}, with entries $y^{(i)}$. You are welcome to slice X and y into the female and male portions, *e.g.*, $X_{\mathrm{f}} = X_{1:n,1:N^{\mathrm{f}}}$ or $y_{\mathrm{m}} = y^{\mathrm{d}}_{N^{\mathrm{f}}+1:N}$.

17 Constrained least squares applications

17.1 *Portfolio optimization.* In this problem you will optimize a set of holdings to minimize risk for various average returns. Download the files `portfolio_data.jl` and `asset_prices.csv` and place them both in your working directory. Include `portfolio_data.jl` in your code, it defines the return matrices `R_train` of dimension `T_train` by `n` and `R_test` of dimension `T_test` by `n` (where `n` is the number of assets and `T_train`, `T_test` are number of days), and the `asset_names` array. (The train data covers the time period 2005-2013, the test data 2013-2015.) Using `R_train` find asset allocation weights for the portfolios that minimize risk for annualized returns of 10% and 20%. (To obtain annualized return multiply the daily return by $P = 250$ trading days.)

Plot the cumulative value for each portfolio over time, starting from the conventional initial investment of \$10000, for both the train and test sets of returns. To compute the product $10^4(1+r_1^Tw)(1+r_2^Tw)\cdots(1+r_T^Tw)$ you can use `1e4*cumprod(1+returns*w)` where `returns` is a returns matrix and `w` an allocation vector.

For each of the 2 portfolios report

- the annualized return on the training and test sets;
- the annualized risk on the training and test sets;
- the asset with the minimum allocation weight (can be the most negative), and its weight;
- the asset with the maximum allocation weight, and its weight;
- the leverage, defined as $|w_1| + \cdots + |w_n|$. (Several other definitions of leverage are used.) This number is always at least one, and it is exactly one only if the portfolio has no short positions. You can use `sum(abs(w))`, where `w` is your allocation vector.

Comment briefly.

17.2 *Rendezvous.* The dynamics of two vehicles, at sampling times $t = 1, 2, \ldots$, are given by

$$x_{t+1} = Ax_t + Bu_t, \qquad z_{t+1} = Az_t + Bv_t$$

where the n-vectors x_t and z_t are the states of vehicles 1 and 2, and the m-vectors u_t and v_t are the inputs of vehicles 1 and 2. The $n \times n$ matrix A and the $n \times m$ matrix B are known.

The position of vehicle 1 at time t is given by Cx_t, where C is a known $2 \times n$ matrix. Similarly, the position of vehicle 2 at time t is given by Cz_t.

The initial states of the two vehicles are fixed and given:

$$x_1 = x_{\text{start}}, \qquad z_1 = z_{\text{start}}.$$

We are interested in finding a sequence of inputs for the two vehicles over the time interval $t = 1, \ldots, T-1$ so that they *rendezvous* at time $t = T$, *i.e.*, $x_T = z_T$. You can select the inputs to the two vehicles,

$$u_1, u_2, \ldots, u_{T-1}, \qquad v_1, v_2, \ldots, v_{T-1}.$$

Among choices of the sequences $u_1, \ldots, u_{T-1}$ and $v_1, \ldots, v_{T-1}$ that satisfy the rendezvous condition, we want the one that minimizes the weighted sum of squares of the two vehicle inputs,

$$J = \sum_{t=1}^{T-1} \|u_t\|^2 + \lambda \sum_{t=1}^{T-1} \|v_t\|^2,$$

where $\lambda > 0$ is a parameter that trades off the two objectives.

(a) Explain how to find the sequences $u_1, \ldots, u_{T-1}$ and $v_1, \ldots, v_{T-1}$ that minimize J while satisfying the rendezvous condition by solving a constrained least-squares problem.

(b) The problem data A, B, C, x_{start}, and z_{start} are defined in `rendezvous.jl`. Use `LinearLeastSquares` to find $u_1, \ldots, u_{T-1}$ and $v_1, \ldots, v_{T-1}$ for $\lambda = 0.1$, $\lambda = 1$, and $\lambda = 10$. Plot the vehicle trajectories (*i.e.*, their positions) for each λ using the plotting code in `rendezvous.jl`.

(c) Give a simple expression for x_T in the limit where $\lambda \to \infty$ and for z_T in the limit where $\lambda \to 0$. Assume that for any $w \in \mathbf{R}^n$ there exist sequences $u_1, \ldots, u_{T-1}$ and $v_1, \ldots, v_{T-1}$ such that the rendezvous constraints are satisfied with $w = z_T = x_T$.

17.3 *A linear regulator for a linear dynamical system.* 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.*

(a) 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 control is called the *minimum energy regulator.*

Hint. Your formula may involve the *controllability matrix*

$$C = \left[\begin{array}{cccc} B & AB & \cdots & A^{T-2}B \end{array}\right],$$

and the vector $u = (u_{T-1}, u_{T-2}, \ldots, u_1)$ (which is the input sequence in reverse order). You do not need to expand expressions involving C, such as CC^T or $C^\dagger$, in terms of A and B; you are welcome to simply give your answers using C. You may assume that C is wide and has independent rows.

(b) Show that u_t (the tth input in the sequence found in part (a)) can be expressed as $u_t = K_t x_1$, where K_t is an $m \times n$ matrix. Show how to find K_t from A and B. (But feel free to use the matrix C in your answer.)

Hint. Your expression for K_t can include submatrices of C or $C^\dagger$.

(c) *A constant linear regulator.* A very common regulator strategy is to simply use $u_t = K_1 x_t$ for all t, $t = 1, 2, 3, \ldots$. This is called a (constant) *linear regulator*, and K_1 is called the *state feedback gain* (since it maps the current state into the control input). Using this control strategy can be interpreted as always carrying out the first step of minimum energy control, as if we were going to steer the state to zero T steps in the future. This choice of input does not yield regulation in T steps, but it typically achieves *asymptotic regulation*, which means that $x_t \to 0$ as $t \to \infty$.

Find the state feedback gain K_1 for the specific system with

$$A = \left[\begin{array}{ccc} 1.003 & 0 & -0.008 \\ 0.005 & .997 & 0 \\ 0 & 0.005 & 1.002 \end{array}\right], \qquad B = \left[\begin{array}{cc} 1 & 0 \\ 4 & 5 \\ 6 & 2 \end{array}\right],$$

using $T = 10$. You may find the code in `regulation.jl` useful.

(d) Simulate the system given in part (c) from several choices of initial state x_1, under two conditions: *open-loop*, which means $u_t = 0$, and *closed-loop*, which means $u_t = K_1 x_t$, where K_1 is the state feedback gain found in part (c). Use `regulation.jl` to plot x.

17.4 *Index tracking via least squares.* An *index* is a real or hypothetical portfolio, such as the S&P 500, Dow Jones Industrial Average, or the NASDAQ composite. Here we focus on a generic index, with returns over T trading days given by the T-vector r, with r_t the index return on day t. In *index tracking*, we create a portfolio of n assets whose return is close to (or tracks) the index return. We consider a fixed portfolio of n assets, given by weight n-vector w, with $\mathbf{1}^T w = 1$. (The weight vector can have negative entries, corresponding to short positions.) The returns of the n assets over the T trading days is given by the $T \times n$ return matrix R, with R_{tj} being the return of asset j in trading day t. The return of the portfolio is the T-vector Rw. We will assume that the columns of R are linearly independent (which implies that $T \geq n$).

In *least squares index tracking*, we pick w to minimize the RMS deviation between the index return r and the portfolio return Rw, subject to $\mathbf{1}^T w = 1$.

Give a formula for the optimal portfolio weight vector $\hat{w}$. Your formula can involve R, r, block matrices and vectors, slicing, inverses, transpose, and other standard matrix operations. If your method involves an inverse (or multiple inverses), explain why the matrix (or matrices) being inverted is (are) invertible.

Remark. This exercise differs a bit from the way index tracking is really done in practice. In practical index tracking it is common to change the portfolio weights every day, but here we consider the case when the portfolio weights are fixed. In addition the least squares problem typically uses weights that emphasize recent returns over returns from the past, such as exponentially decaying weights. (You don't need to know any of this to solve this problem.)

18 Nonlinear least squares

18.1 *Airplane steady flight.* Consider an airplane flying at constant speed S, along a straight line with *flight path angle* γ (in radians, with respect to horizontal; $\gamma > 0$ means climbing). The other relevant quantities are α, the *angle of attack*, which is the angle between the airplane body and the airplane flight path, the four forces *lift* L, *drag* D, engine *thrust* T, and airplane weight W. These are related by the *steady flight equations*,

$$T = W\gamma + D, \qquad L = W$$

which state that the horizontal and vertical forces on the airplane balance, *i.e.*, sum to zero. The lift and drag forces depend on the angle of attack and speed,

$$L = a_L C_L(\alpha) S^2, \qquad D = a_D C_D(\alpha) S^2,$$

where a_L and a_D are known constants, C_L is the *coefficient of lift function*, and C_D is the *coefficient of drag function.* These are well approximated as affine functions,

$$C_L(\alpha) = C_L^{(0)} + \alpha C_L^{(1)}, \qquad C_D(\alpha) = C_D^{(0)} + \alpha C_D^{(1)},$$

where the four constants $C_L^{(0)}, C_L^{(1)}, C_D^{(0)}, C_D^{(1)}$ are known. The airplane weight W is known and constant; we can control the engine thrust T and the angle of attack α (via the airplane control surfaces).

(a) Given a desired speed S^{des} and a desired flight path angle γ^{des}, how would you determine the thrust T and angle of attack α. Explain how this can be done analytically. (You do not have to give formulas for T and α.)

(b) *Glide path.* Suppose that $T = 0$, *i.e.*, the engines are off. the flight path angle γ and the speed S.

(c) Now suppose that we fix T and α. Use the Levenberg-Marquardt algorithm to compute the resulting speed S and flight path angle γ.

18.2 *Lambert W-function.* The *Lambert W-function*, denoted $W : [0, \infty) \to \mathbf{R}$, is defined as $z = W(u)$, where z is the unique number $z \geq 0$ for which $ze^z = u$. (The notation just means that we restrict the argument z to be nonnegative.) The Lambert function arises in a variety of applications. 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 $(ze^z - u)^2$ over z.

(a) Suppose the current iterate is $z^{(k)}$ (which you can assume to be nonnegative). Give a formula for $z^{(k+1)}$ in terms of $z^{(k)}$ and $\lambda^{(k)}$, the current value of the regularization parameter in the Levenberg-Marquardt algorithm.

(b) Implement the Levenberg-Marquardt algorithm. You can start with $z^{(1)} = 1$ and $\lambda^{(1)} = 1$ (but it should work with other initializations). You can stop the algorithm when $\left| z^{(k)} e^{z^{(k)}} - u \right|$ is small (say, less than 10^{-6}). Test your algorithm by evaluating $W(1)$, $W(2)$, and $W(3)$. Verify that these numbers (almost) satisfy $W(i)e^{W(i)} = i$, for $i = 1, 2, 3$. If you like, you can check the values you compute against those computed using the `LambertW` Julia package.

19 Constrained nonlinear least squares

19.1

20 Miscellaneous

20.1 *Appropriate response.* For each of the situations (a), (b), and (c) described below, circle the most appropriate response from among the five choices. You can circle only one for each situation.

(a) An intern working for you develops several different models to predict the daily demand for a product. How should you choose which model is the best one, the one to put into production?

Laplacian *Magic* *Regularization* *Validation* *k-means*

(b) As an intern you develop an auto-regressive model to predict tomorrow's sales volume. It works very well, making predictions that are typically within 5% of the actual sales volume. Your boss, who has an MBA and is not particularly interested in mathematical details, asks how your predictor works. How do you respond?

Laplacian *Magic* *Regularization* *Validation* *k-means*

(c) A colleague needs a quantitative measure of how rough an image is, *i.e.*, how much adjacent pixel values differ. What do you suggest?

Laplacian *Magic* *Regularization* *Validation* *k-means*