Quantitative Engineering Analysis I

Fifth Edition

Spring 2020

Contents

Module I

Faces: Linear Algebra Through Facial Recognition

Chapter 1

Day 1: Facial Recognition: The Big Picture

1.1 Schedule

- 0900-0920: Welcome - A letter to my future self
- 0920-0950: A simple face detection: Round 1
- 0950-1010: Round 2: What did you notice?
- 1010-1030: Debrief: What did we learn?
- 1030-1045: Coffee
- 1045-1055: Pixel arithmetic
- 1055-1105: A Universal Set of Building Blocks
- 1105-1125: A Better Set of Building Blocks?
- 1125-1145: Towards an Optimal Basis
- 1145-1200: Broadcast debrief (via Zoom)
- 1200-1220: Course logistics
- 1220-1230: Day 1 survey

Welcome to QEA Module One! In this module, you will develop software to recognize your face among everyone in QEA (hello, new late-night security). It all functions through applying some beautiful mathematics and using computational tools. Let's first imagine how a computer "sees" an image as numbers.

1.2 Facial recognition- "seeing" via numbers

Round 1: From images to numbers [30 mins]

At your table you will find a smiley face. Imagine converting this face into a form that a computer can understand (i.e., numbers). A grid is superimposed on the face for your reference.

Goal Design a method that enables a "computer" to (approximately) reproduce the face from a list of numbers and an algorithm that you define. The numbers can be grouped within the list, but your list should contain numbers only. An example of a group of numbers is [2,4] or [0, 100, 14]. You will create the algorithm (or, equivalently, the instructions) that tell the computer what to do with your list of numbers.

When you've defined your group's method,

- Generate the list of numbers that represents your face using your method.
- Make a set of instructions (your "algorithm") on your portable white board using a BLACK marker so that another group can recreate your image from your list of numbers.

- Trade instructions with another group.
- Create the other group's face from their algorithm on the blank grid.
- Record any challenges you encounter on their portable whiteboard using a RED marker.
- Exchange back your original materials and debrief on what you've learned about your method at your table.

Round 2 [20 mins]

Goal Adjust your method to be able to distinguish the new faces that you've just been given. The 8x8 grid is shown for reference; you are not restricted to this grid.

Discuss the following and record your answers on your portable whiteboard:

- How does your method need to transform the original image in order to "see" the detail of the face?
- What "demands" does your new method make on the computer compared to the old method? (Remember that the computer is using your algorithm and numbers to represent the images)
- Consider a photo of a human face, in what ways does your numerical method contain inherent limits or biases?

A debrief (in each room) [20 mins]

Coffee Break [15 min]

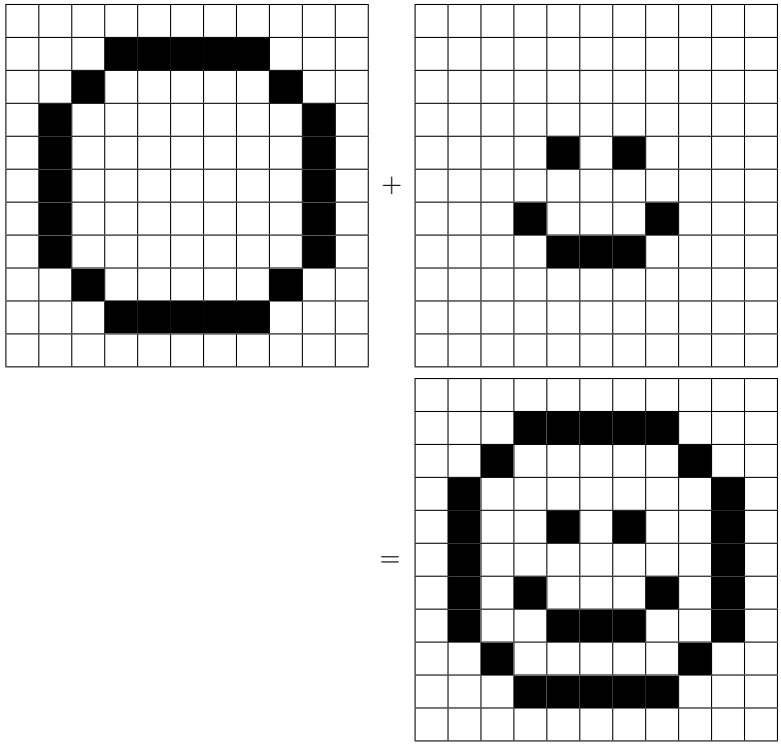
1.3 Facespace

Before the coffee break we thought about various ways to represent an image (e.g., a picture of a face). In this section we're going to narrow in on a particular method of representing images: as a weighted sum of a set of building block images. In this section you'll work through some exercises to scaffold the basic ideas of how this type of representation works and why it is so powerful.

Pixel Arithmetic [10 mins]

Adding is one of the most basic operations in mathematics. While everyone here is familiar with the concept of adding numbers, we can generalize this idea to add together other sorts of entities. We can even think about what it means to add two images together.

As a simple example, let's add the following two images together (we'll explain more precisely how we are defining addition of images once you've seen the result).



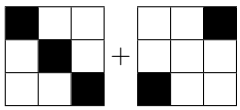
Conceptually, this operation might seem straightforward. Adding two images results in an image that has a black pixel whenever either of the two images has a black pixel at a corresponding position.

More formally, we can think about black pixels as having a value of 255 and white pixels as having a value of 0 (gray pixels would have a value between these two values depending on how dark they are). (A scale from 0 to 255 seems like a weird choice, but there is a very good reason why this is the standard - remember that digital storage uses binary (bit) - how many integers can you represent with an 8-bit number?) To add two images together, all we do is add the corresponding elements at a particular point in the grid! In this way addition on images works much the same as addition of a single number–the only difference is we perform the addition of single numbers multiple times for each position in the grid.

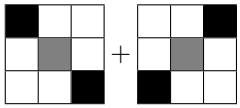
Exercise 1.1

With your tablemates, work through the following pixel arithmetic problems on the board.

1.

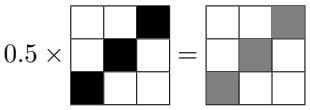


2.



Without too much of a leap, we can also multiply images by a number by simply multiplying each element in the image by that value. We can think of this multiplication operation as "scaling" the image.

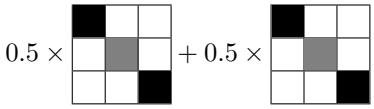
For example,



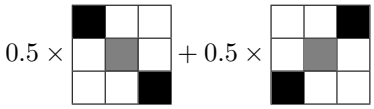
Exercise 1.2

With your tablemates, work through the following pixel arithmetic problems on the board.

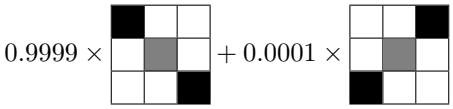
1.



2.



3. (Don't think about this one too hard. Just draw approximately what this would be)



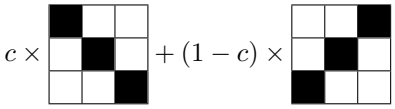
A Universal Set of Building Block Images [10 mins]

Now that we have a sense of how we can add and scale images, let's think about how we might construct a set of building block images such that we can construct any image as a sum of scaled versions of these building blocks.

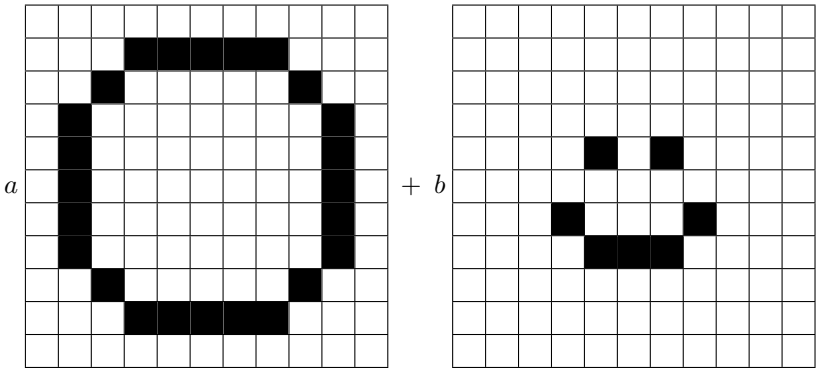
Exercise 1.3

With your tablemates, work through the following problems.

1. What is the range of images that could be constructed by summing over scaled versions of the following building block images? (c is a number between 0 and 1). Another way to think about this is, as you sweep the value of c from 0 to 1, how does the resultant sum of the two images change?



2. What is the range of images that could be constructed by summing over scaled versions of the following building block images? (a and b are both numbers between 0 and 1). Instead of having one knob to turn (as in the previous exercise), you now have two.



   In this case we can think of the values a and b as encoding of a particular smiley face. You will deduce the effect that both a and b have on the specific nature of the smiley face.

3. Building on the previous example, come up with your own way of representing a simple face like the one above as the sum of two or more scaled building block images. This is intended to be fun, so be creative! It's up to you what sort of faces that your method is capable of representing.

4. You probably noticed from the previous three exercises that not all possible images can be constructed by adding scaled versions from a small set of building block images. Suppose you wanted to be able to represent *any* possible 3 pixel by 3 pixel image of a face. While there are many possible ways to do this, for simplicity each of your building block images should only have a single black pixel (the rest should be white). At the board, define a set of building block images that lets you represent any possible 3 pixel by 3 pixel face in this manner. How many building block images did you need to represent all possible 3 pixel by 3 pixel faces?

Are there any images that can't be represented as a sum of scaled versions from your building block images? How many building block images would you need if you wanted to encode all possible 5 pixel by 5 pixel faces? What about n pixels by n pixels?

A Better Set of Building Blocks? [20 mins]

At the end of the previous section you showed how can represent any possible image as a sum of scaled single-pixel images. This is a very powerful idea, but we can take it even farther. Before we continue, let's think about some of the ways in which this way of representing face images is not so great.

Exercise 1.4

Suppose you wish to represent 19 pixel by 19 pixel images of faces using the scheme you devised in the previous set of exercises (as a sum of scaled, single-pixel images). Here is an example of what such a face might look like.

1. If you think of the representation of each image as the scaling factor that you apply to each of your single-pixel images, how many numbers do you need to specify this one face image (you answered almost this exact question in the previous part, so don't overthink this).
2. How many numbers would you need to represent a 19 pixel by 19 pixel image of a flower? How many numbers would you need to represent a completely random 19 pixel by 19 images (one with no special structure)?
3. Suppose someone gives you one of the numbers needed to encode a particular face? Without looking at the face image itself, how much information (e.g., age, identity, sex, gender, etc.) could you determine about the face just from that one number?

As you probably deduced in the previous exercise, a major drawback of the encoding we worked out previously is that each scaling factor doesn't really tell us that much useful information about each face (and as a result we need a lot of these numbers to specify a particular face). It turns out that we can fix a lot of these shortcomings through more carefully choosing our set of building block images. Reframing problems by choosing a different set of building blocks is going to be one of the key ideas in this module.

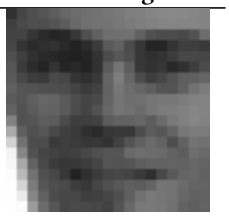
Come to the front of the room and grab a piece of paper with a 6 by 4 grid of face-like images along with a set of transparent face-like images held by a binder clip. Take these materials back to your table. Layout

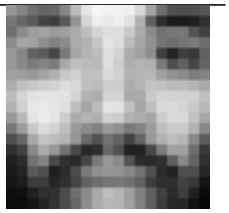
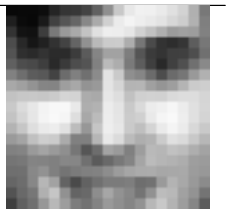
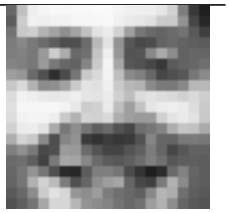
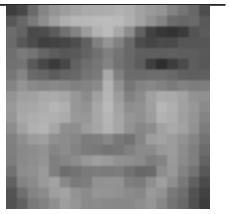
the piece of paper on your table. Also in the envelope you should have a set of transparent versions of those same building block images. Layout the transparent building block images so that they align with the appropriate printed building block image. The bottom building block should go in the upper left corner of the printed sheet. As a sanity check, make sure the textured side of the transparency is facing up (one side will be smooth and the other textured). Be very careful when laying out your images as it is hard to get them back in the right order.

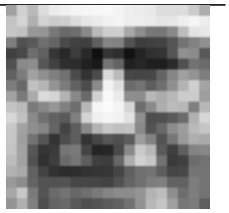
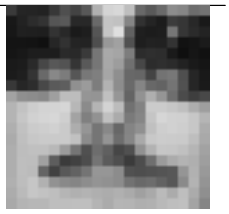
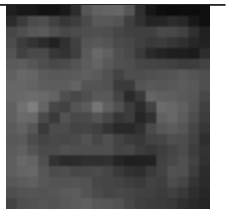
What you see before you is a very carefully chosen set of building block images. You should notice that each row represents a different building block image and each column represents a different scaled version of that same building block. Today, we won't be going into detail about *how* we determined these particular building blocks but we will be having you experiment with them in order to understand, at a conceptual level, some of their properties.

- You can add these scaled building block images by simply stacking multiple transparencies on top of each other and placing them on a white background (make sure to keep them aligned). We've found that using your thumb and index finger and pinching the middle of the transparency is a good way to pick it up (they are pretty sturdy).
- Along with these building block images, we have determined optimal encodings for a bunch of different faces. At your table, pick a few of these faces and try assembling them (you should probably put the transparencies back after assembling each face so you can keep better track of the transparencies).

  Note: that each column in the table corresponds to one of the building block images (row of your transparencies). Higher numbers in the table correspond to choosing the darker (more saturated) versions of each building block image. If a 0 appears for a particular building block, don't include that building block at all to construct a particular face.

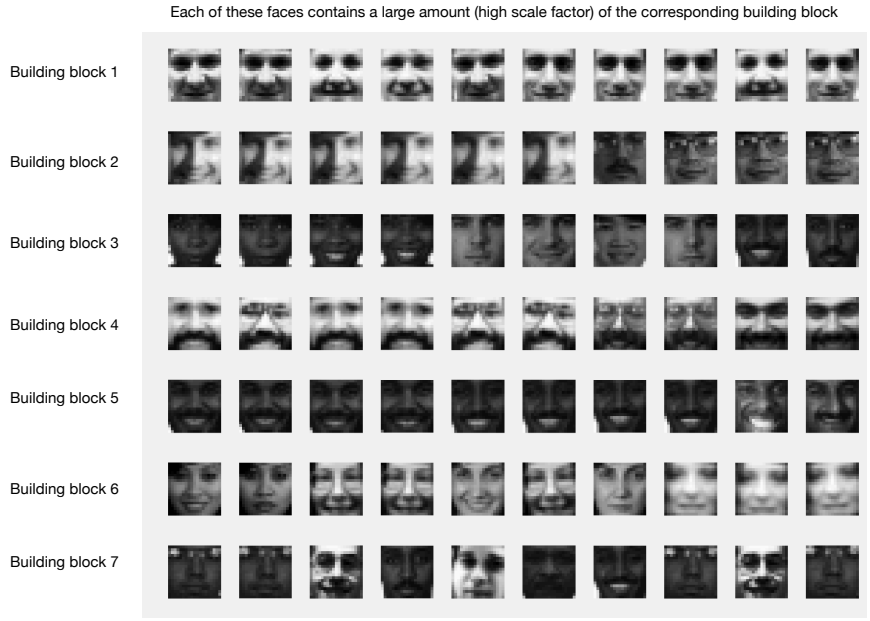
Intensity 1	Intensity 2	Intensity 3	Intensity 4	Intensity 5	Intensity 6	face image
3	3	0	0	2	2	
0	2	3	2	3	1	
3	0	1	4	1	1	

1	4	1	2	3	4	
0	0	3	3	2	1	
2	0	3	0	2	0	
2	0	1	3	0	1	
1	2	0	1	0	0	
1	1	3	0	1	2	

2	0	1	3	0	1	
3	2	0	2	0	1	
2	1	3	0	3	1	
0	3	0	1	3	3	
2	3	0	1	2	1	

- How many numbers do you now need to encode a 19 pixel by 19 pixel face?
- Can you encode any possible face with this set of building blocks?
- How well does this set of building blocks work for encoding these faces? Does it seem to work equally well across all faces? Which faces does it work well on (i.e., they can accurately be reconstructed from the building blocks) and which faces does it work poorly on?
- Looking at the building blocks themselves, what does each building block seem to represent? In other words, as you increases the amount of a particular building block, what features or qualities does that impart on the resulting face. To help you think this through, below we have a grid of faces where

each row corresponds with one of the six building block images and each of the faces in the row contains a large amount of that particular building block image in its encoding.



Towards an Optimal Basis [20 mins]

Exercise 1.5

In this question, we want you to think about process rather than particular techniques for solving this problem. If you have questions on what we mean by this, let us know.
Suppose someone has hired you as a consultant to create a method to encode 19 pixel by19 pixel images of faces (similar to the ones you just experimented with) as a sum of scaled versions of just 10 building block images.

1. What questions would you want to ask the person that hired you in order to do a good job on this project? (i.e., what information do you need to know?)
2. What might be some qualities of a good set of building block images? (e.g., how would they look? what sort of dimensions of variability would they have?)
3. What sort of data might you need to collect in order to inform the set of building blocks you will ultimately deliver (this data could be images or it could be other quantitative or qualitative data)?
4. How might you determine whether your method is working (these could be quantitative measurements or qualitative observations of your system)?

5. Are there any other steps might you want to take to complete the project?

6. We will be digging into the various dimensions of the use of facial recognition technology in society later in this module, but for now we want to get you thinking about two particular components of that. Many face processing technologies work best on white males (e.g., check out the Gender Shades project). One possible explanation for this phenomenon is overt bias on the part of the creators of these technologies. Instead, for the sake of this exercise, let's suppose that the differences in performance are actually the result of subtle, unconscious bias in any number of decisions that the technology creators made during the design process. A second problem that plagues face processing algorithms is that they seem to work great when evaluated in the settings that the technology designers had in mind when they built the technology, but often work poorly when deployed in the real world. Looking back on the steps you listed above, flag steps that might have the potential to introduce bias into your system (e.g., having your system work better on one group of people than another or having it fail in a particular use case). It's okay if you don't know where bias might creep in, the purpose of this exercise is to get you asking questions rather than reaching conclusions.

Chapter 2

Night 1: Introduction to Matrices

Overview and Orientation

In this night assignment, we will learn some of the foundational material about matrices and matrix operations.

Learning Objectives

Concepts

- Define a vector, a matrix and an array
- Describe the meaning of the dimensions of a vector, a matrix, and an array
- Give at least one interpretation of matrix-vector multiplication
- Calculate the product of a matrix-vector multiplication for 2D and 3D matrices
- Understand dimensionality-requirements for matrix-vector multiplication and predict resulting dimensions
- Define and recognize the following special matrices: Identity, diagonal, square, rectangular, symmetric

MATLAB skills

- Determine the dimensions of a vector, matrix, or array variable
- Perform operations (addition, multiplication, transposition) on matrices
- Extract desired subarrays or matrices from arrays

Suggested Approach

- First you should quickly scan through the assignment, see what is being asked, and assess the extent to which you already know how to do things. Spend no more than 30 minutes or so doing this.
- You should then read the assignment more closely, try out problems, and if appropriate, look at some of the other resources that are suggested. Don't spend more than 1 hour poking around at stuff online unless it is really being productive: it's easy to spend a lot of time there without accomplishing much.
- Then start doing the problems in earnest, and/or spend focused time with suggested resources.
- Once you've spent a total of 3-4 hours working on the assignment, you should check your progress. Are you on track to finish within about 7-8 hours? Do you feel confident that you can do the stuff that's left? If not, this is when you should ask for help. This means talk to a colleague, or talk to a ninja, or track down an instructor, or send an email to an instructor.

- You should turn in a PDF document with answers to all the numbered questions below. For the MATLAB assignments, please export your work to pdf. Please carefully label the problem number in your MATLAB script.

Resources to read and watch

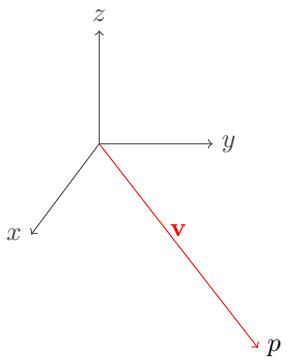
There are lots of books about Linear Algebra and lots of useful videos on the web. Here are some specific recommendations:

- Introduction to Linear Algebra, by Strang
- Linear Algebra, by Lay
- Linear Algebra, by Cherney, Denton, Thomas, Waldron
- Homebrew videos
  - Matrices operating on vectors
  - Matrices operating on vectors (example)
  - Matrices operating on matrices
- Videos from others
  - Vectors, the very basics
  - 3Blue1Brown's YouTube series on Linear Algebra

2.1 *Linear Algebra, Vectors, and Matrices*

In your concept-maps for eigenfaces, most, if not all of you would have included something about linear algebra, matrices, and vectors. These topics are used very heavily in many different areas, including in data analysis. For the next couple of weeks, you will spend a good deal of time learning about these things and how to apply them.

Linear Algebra and Vectors



Consider the point $p = (1, 2, -1)$ in 3-dimensional space. We can associate a position vector $\mathbf{v}$ with this point, which is the vector from the origin to this point,

$$\mathbf{v} = \begin{bmatrix} 1 \\ 2 \\ -1 \end{bmatrix}.$$

Likewise, we can think of every vector as defining a point, if we assume that the vector emanates from the origin. So, for example, the vector

$$\mathbf{v} = \begin{bmatrix} 3 \\ -2 \\ 0 \\ 1 \end{bmatrix}$$

is identified with the point $(3, -2, 0, 1)$ in 4D. Often times we will mix and match these ideas and say things like: the vector (x, y, z). What we really mean when we say this is: the point (x, y, z) can be treated as the position vector

$$\mathbf{v} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

The vector $\mathbf{v}$, as represented above, is called a column vector. We can also have row vectors such as the following

$$\mathbf{u} = \begin{bmatrix} p & q & r \end{bmatrix}.$$

The operation of converting a column vector to a row vector or vice-versa is called taking the *transpose* of the vector and is denoted with a superscript T. For example, the transpose of the row vector $\mathbf{u}$ from above is

$$\mathbf{u}^T = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{2.1}$$

and the transpose of the vector $\mathbf{v}$ from above is

$$\mathbf{v}^T = \begin{bmatrix} x & y & z \end{bmatrix}. \tag{2.2}$$

We can take the product of a row vector with a column vector using the following formula

$$\mathbf{uv} = \begin{bmatrix} p & q & r \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = px + qy + zr \tag{2.3}$$

If we start with two column vectors

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \text{ and } \mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$$

of length n (i.e., they are n=dimensional), then we can take the *d*ot product

$$\mathbf{v} \cdot \mathbf{w} = v_1 w_1 + v_2 w_2 + \cdots + v_n w_n.$$

In some sense, the dot product is a measure of how aligned two vectors are. Here's the key formula:

$$\mathbf{v} \cdot \mathbf{w} = \|\mathbf{v}\| \|\mathbf{w}\| \cos\theta$$

where θ is the angle between $\mathbf{v}$ and $\mathbf{w}$ and

$$\|\mathbf{v}\| = \sqrt{v_1^2 + v_2^2 + \cdots + v_n^2}^{1/2}$$

is the length of the vector $\mathbf{v}$ in n-dimensional space.

Exercise 2.1

1. Assume $\mathbf{v}$ and $\mathbf{w}$ are two vectors of unit length, i.e., $\|\mathbf{v}\| = \|\mathbf{w}\| = 1$. Using the formula above, what angle between $\mathbf{v}$ and $\mathbf{w}$ maximizes the dot product? Using the formula above, what angle between $\mathbf{v}$ and $\mathbf{w}$ minimizes the dot product?
2. Compute $\mathbf{v} \cdot \mathbf{w}$ where

$$\mathbf{v} = \begin{bmatrix} 1 \\ 3 \\ -4 \\ 6 \end{bmatrix}, \text{ and } \mathbf{w} = \begin{bmatrix} -2 \\ 0 \\ 1 \\ 3 \end{bmatrix}$$

We'll learn more about the dot product as we go. For now, notice that the dot product equals the product of the transpose of one with the other

$$\mathbf{v} \cdot \mathbf{w} = \mathbf{v}^T \mathbf{w}. \tag{2.4}$$

Vectors can also be used to represent many things, such as data. Linear algebra provides a powerful set of tools to manipulate and analyze this data.

Exercise 2.2

For instance, you may have a three-dimensional vector $\mathbf{f}$ whose entries represent the numbers of different fruits you have in your refrigerator. For example, the first entry could be the number of oranges, the second the number of grapefruits and the third could be the number of apples. When organized in this manner, you can use products of row and column vectors to compute the number of different fruits there are. For instance, suppose that

$$\mathbf{f} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \tag{2.5}$$

i.e. you have 1 orange, 2 grapefruits, and 3 apples in your fridge.

1. Find a row vector $\mathbf{t}$ so that the product $\mathbf{tf}$ tells you the total number of fruits in your refrigerator.
2. Find a row vector $\mathbf{c}$ such that the product $\mathbf{cf}$ tells you the total number of *citrus* fruits in your refrigerator.
3. Suppose that in the genetically engineered future, all apples weigh 100 g, all grapefruits weigh 250 g and all oranges weigh 120 g. Find a row vector $\mathbf{w}$, such that the product $\mathbf{wf}$ tells you the total weight of fruits in your refrigerator.

If you wanted to know the vitamin C content of the fruits in your fridge, you could formulate a similar vector to compute it.

In the questions above, you took *linear combinations* of the entries of the vector $\mathbf{f}$ which gave you the desired quantity. *Linear algebra is the study of linear functions.*

Introduction to matrices

Matrices are a set of numbers organized in a two-dimensional array. Matrices are a compact way to represent linear combinations. Matrices can also be used in a number of different ways, such as to represent data. When we multiply a matrix by a vector, it results in a new vector. Therefore, when we say "a matrix operates on a vector", we mean that the matrix multiplies the vector. Notation-wise, we use bold upper-case letters, e.g. $\mathbf{A}$, to represent a matrix and bold lower-case letters to represent a vector, e.g. $\mathbf{v}$.

For instance, you may define a two-dimensional matrix $\mathbf{G}$ with two rows and three columns as follows

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}. \tag{2.6}$$

Matrices and vectors come in different shapes and sizes and we refer to their shape and size by the number of rows and columns they have. A general matrix $\mathbf{A}$ has m rows and n columns, and we refer to this as an $m \times n$ matrix. Vectors are then examples of matrices: row vectors have a single row, i.e., they are $1 \times n$ matrices; and column vectors have a single column, i.e., they are $m \times 1$ matrices.

Matrices can only multiply vectors of a certain size and produce vectors of a certain size: an $m \times n$ matrix can only operate on a column vector of size $n \times 1$, and will produce an output vector which is a column vector of size $m \times 1$. (Likewise, matrices can only multiply other matrices of a certain size: an $m \times n$ matrix can only act on a matrix of size $n \times k$, and will produce an output matrix of size $m \times k$.) These basic properties will become clearer when we look at an example.

Consider the 3×2 matrix $\mathbf{A}$,

$$\mathbf{A} = \begin{bmatrix} 2 & 1 \\ 3 & -1 \\ 0 & 4 \end{bmatrix}$$

and the input vector $\mathbf{v}$

$$\mathbf{v} = \begin{bmatrix} -2 \\ 1 \end{bmatrix}.$$

The output vector $\mathbf{w}$ is computed as follows

$$\mathbf{w} = \begin{bmatrix} 2 & 1 \\ 3 & -1 \\ 0 & 4 \end{bmatrix} \begin{bmatrix} -2 \\ 1 \end{bmatrix} = \begin{bmatrix} (2)(-2) + (1)(1) \\ (3)(-2) + (-1)(1) \\ (0)(-2) + (4)(1) \end{bmatrix} = \begin{bmatrix} -3 \\ -7 \\ 4 \end{bmatrix}$$

There are two main ways to think about this multiplication. The most common view is to treat each entry of the new vector as a dot product between a row of the matrix and the column vector. So, for example, the first entry in the output vector is the dot product of two vectors

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} -2 \\ 1 \end{bmatrix} = -3$$

The second approach is to view the output vector as a linear combination of the columns of the matrix. The entries in the original vector are used as multiplication weights on each column of the matrix, i.e.

$$(-2) \begin{bmatrix} 2 \\ 3 \\ 0 \end{bmatrix} + (1) \begin{bmatrix} 1 \\ -1 \\ 4 \end{bmatrix} = \begin{bmatrix} -3 \\ -7 \\ 4 \end{bmatrix}$$

We encourage you to use both approaches when you think about multiplication.

Exercise 2.3

Recall the matrix $\mathbf{G}$ defined in equation (2.6) and the vector $\mathbf{f}$ defined in Exercise 2.2, which kept track of the number of fruit of different types. What does the vector $\mathbf{Gf}$ represent?

Exercise 2.4

If a matrix multiplies a spatial vector, the resulting vector is *transformed* by the matrix, resulting in a new vector.

1. Please draw the spatial vector

$$\mathbf{v} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{2.7}$$

2. Please draw the vector $\mathbf{w} = \mathbf{A}\mathbf{v}$, where $\mathbf{A}$ is

$$\mathbf{A} = \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \tag{2.8}$$

3. What happened to $\mathbf{v}$ when you multiplied by $\mathbf{A}$?
4. Please draw the vector $\mathbf{u} = \mathbf{B}\mathbf{v}$, where $\mathbf{B}$ is

$$\mathbf{B} = \begin{bmatrix} \cos(30°) & -\sin(30°) \\ \sin(30°) & \cos(30°) \end{bmatrix} \tag{2.9}$$

5. What happened to $\mathbf{v}$ when you multiplied by $\mathbf{B}$?
6. Please draw the vector $\mathbf{t} = \mathbf{R}\mathbf{v}$, where $\mathbf{R}$ is

$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{2.10}$$

7. What happened to $\mathbf{v}$ when you multiplied by $\mathbf{R}$?
8. Please draw a new spatial vector

$$\mathbf{w} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{2.11}$$

9. Please draw the vector $\mathbf{s} = \mathbf{R}\mathbf{w}$
10. What does multiplying *any* vector by $\mathbf{R}$ do?

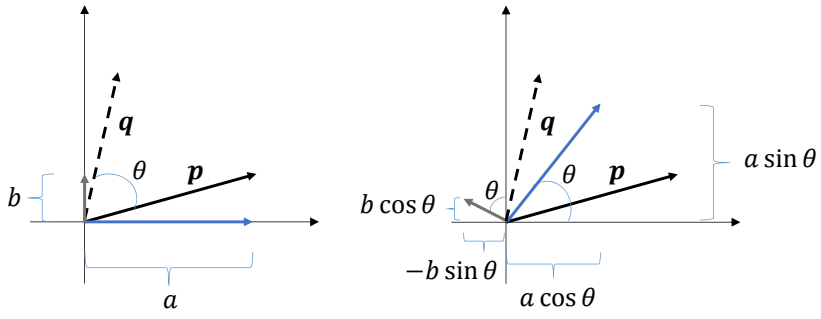


Figure 2.1: Rotation of vectors

You may have guessed that $\mathbf{R}$ defined above, rotates a vector counter-clockwise by θ. This is indeed true, and $\mathbf{R}$ is called a *rotation matrix* as it transforms vectors by rotating them. To understand why $\mathbf{R}$ is a rotation matrix, consider Figure 2.1 (i). Suppose that we wish to rotate the vector $\mathbf{p}$ counter-clockwise by θ, which will result in the vector $\mathbf{q}$. From the figure, we see that

$$\mathbf{p} = \begin{bmatrix} a \\ b \end{bmatrix}, \tag{2.12}$$

and $\mathbf{p}$ is the sum of the gray and blue vectors. If we now rotate the blue and gray vectors counter-clockwise by θ, we see that $\mathbf{q}$ is the sum of the rotated versions of the blue and gray vectors, as shown in Figure 2.1 (ii). By using trigonometry, we see that the blue vector in Figure 2.1 (ii) is

$$\begin{pmatrix} a\cos\theta \\ a\sin\theta \end{pmatrix} \tag{2.13}$$

and the gray vector in Figure 2.1 (ii) is

$$\begin{pmatrix} -b\sin\theta \\ b\cos\theta \end{pmatrix} \tag{2.14}$$

Therefore, $\mathbf{q}$ is given by

$$\mathbf{q} = \begin{pmatrix} a\cos\theta \\ a\sin\theta \end{pmatrix} + \begin{pmatrix} -b\sin\theta \\ b\cos\theta \end{pmatrix} = + \begin{pmatrix} a\cos\theta - b\sin\theta \\ a\sin\theta + b\cos\theta \end{pmatrix} = \mathbf{R}\mathbf{p}. \tag{2.15}$$

General Notation

As we mentioned earlier, $m \times n$ matrices can multiply $n \times 1$ vectors and produce $m \times 1$ vectors. Consider a generic $m \times n$ matrix $\mathbf{A}$

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \cdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

where the ij-th entry of this matrix, a_{ij} defined above, is the entry corresponding to the i-th row and j-th column. You can multiply an $n \times 1$ vector $\mathbf{v}$ by this matrix. Define the vector $\mathbf{v}$ as follows,

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}.$$

Now define another vector $\mathbf{w}$ which is the product of $\mathbf{A}$ and $\mathbf{v}$, i.e., $\mathbf{w} = \mathbf{A}\mathbf{v}$. If we define

$$\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix}$$

then the i-th entry of $\mathbf{w}$, is given by the following sum

$$w_i = a_{i1}v_1 + a_{i2}v_2 \cdots a_{im}v_m = \sum_{j=1}^{n} a_{ij}v_j$$

Other Matrix operations

Besides multiplication, a number of other operations can be done using matrices including addition, subtraction, inversion, transposition, etc. We will explore more of these and their associated properties in the next section. All of these operations make matrices a very powerful tool in the study of many different systems which can be represented as linear transformations, or combinations.

2.2 Matrix Operations in MATLAB

Exercise 2.5

In the command window, you can type in commands and press enter. Try the following commands and see what they do.

```
1+1
a=1+1
a
% you can start a comment with "%"
b=2;% this will appear as a variable in your workspace, but the semicolon ...
    suppresses the output
c=3,d=4,e=5;% use commas, semicolons, or shift+enter between commands that you ...
    want to execute together
1+2-(3*4/5)^6
clear a
a% should give you an error because a is not defined anymore
clear all
clc% only if you want to clear your workspace!
```

To practice matrix operations, let's define a matrix and some vectors using MATLAB as follows:

```
>> A = [2 1; 3 -1; 0 4]
```

Note that the semi-colon ends a row and begins a new row. You can also use returns between rows—try it! Square brackets enclose the matrix. To define the column vector $\mathbf{v}$ in MATLAB you can type the following command:

```
>>  v = [-2; 1]
```

whilst to define the row vector $\mathbf{u}$ in MATLAB you can type the following command

```
>>  u = [2 -3 1]
```

Notice that in this case each component of the vector is separated by a space - you could also separate them with a comma.

Exercise 2.6

Using the definitions for **A**, **v**, and **u** from above, please predict the output of the following commands and then solve them using MATLAB.

```
A*v
u*A
A(1:2,:)*v
u*A(:,2)
```

For Night 1, you will also need to plot things. In MATLAB, you can use plot(xv,yv) to create a scatter plot.

```
yv=[1 7 4 5 3 9 2 4]
xv=[1 3 4 6 8 9 11 14]
plot(xv,yv)
```

If you want to know more about how to use a function like plot, use "help." Create the plot above, then type "help plot" into the command window and try to change something about your plot, such as using points instead of a line or adding axis labels.

Finally, you need to know how to use a for loop to repeat a set of commands a number of times. Here's an example for loop that makes a vector that's a sequence of squares: (Try it!)

```
for n=1:3% n is the index variable, which counts from 1 to 3 (call it whatever you want)
v(n)=n^2% assigns the nth component of v to the value n^2 and prints out v
% loop repeats, adding 1 to n each time, until i gets to 3
end% needed to end the loop!
```

Exercise 2.7

Write a for loop that creates the following matrix:

```
M_squares=[1 1;2 4;3 9;4 16]
```

2.3 Elementary Matrix Operations, Properties, and Terminology

In this part of the assignment, you will learn a number of basic operations and properties of matrices which can then be used in applications. Admittedly, most of these exercises are a little dry, but they will be useful in the very near future, we promise!

Matrix-Vector Multiply

Here, you will work on examples of matrices multiplying vectors to get yourselves comfortable with matrix operations in MATLAB. First, let's define the matrix $\mathbf{A}$ using MATLAB as follows

```
>> A = [2 1; 3 -1; 0 4]
```

Note that the semi-colon ends a row and begins a new row. To define the column vector $\mathbf{v}$ in MATLAB you can type the following command:

```
>>  v = [-2; 1]
```

whilst to define the row vector $\mathbf{u}$ in MATLAB you can type the following command

```
>>  u = [2 -3 1]
```

Notice that in this case each component of the vector is separated by a space - you could also separate them with a comma.

Exercise 2.8

Using the definitions for $\mathbf{A}$, $\mathbf{v}$, and $\mathbf{u}$ from above, please solve the following using MATLAB. Do the answers match what you expect? (Not all of these may be defined!)

1. `A*v`
2. `u*A`
3. `A*u`
4. `v*A`
5. `A(1:2,:)*v`
6. `u*A(:,2)`
7. `A(:, 2:4)*v`
8. `u*A(1,:)`

Addition, subtraction, scalar multiplication and transpose of matrices

We can add matrices of the same size, and subtract them from one another. Both operations result in matrices of the same size and shape. The addition and subtraction operations are done element-wise. For instance the difference of the two matrices can be calculated as below

$$\mathbf{A} = \begin{bmatrix} 3 & 4 & 1 \\ 3 & 1 & 1 \end{bmatrix} \tag{2.16}$$

$$\mathbf{B} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 2 & 1 \end{bmatrix} \tag{2.17}$$

$$\mathbf{A} - \mathbf{B} = \begin{bmatrix} (3-1) & (4-2) & (1-3) \\ (3-2) & (2-1) & (1-1) \end{bmatrix} \tag{2.18}$$

$$= \begin{bmatrix} 2 & 2 & -2 \\ 1 & -1 & 0 \end{bmatrix} \tag{2.19}$$

Multiplying a matrix by a scalar simply scales each entry of the matrix by the scale factor. For instance

$$3\mathbf{A} = \begin{bmatrix} 9 & 12 & 3 \\ 9 & 3 & 3 \end{bmatrix} \tag{2.20}$$

The transpose of a vector, denoted by the superscript ^{T} turns a column vector into a row vector, and vice versa. For matrices, the transpose replaces the rows with the columns (or vice-versa). For example,

$$\begin{bmatrix} 1 & 3 & 5 \\ 2 & 7 & 6 \end{bmatrix}^T = \begin{bmatrix} 1 & 2 \\ 3 & 7 \\ 5 & 6 \end{bmatrix} \tag{2.21}$$

Since the columns are replaced with the rows, the shape of the matrix changes when you transpose it. The following property of transposes will be useful moving forward. Consider a matrix $\mathbf{A}$ and a vector $\mathbf{v}$. Then

$$(\mathbf{A}\mathbf{v})^T = \mathbf{v}^T\mathbf{A}^T \tag{2.22}$$

Exercise 2.9

Using $\mathbf{A}$ and $\mathbf{B}$ previously defined, evaluate $4\mathbf{A} - 5\mathbf{B}$

Exercise 2.10

If the matrix $\mathbf{A}$ has dimensions of 4×5, what are the dimensions of $\mathbf{A}^T$?

Exercise 2.11

If the matrix $\mathbf{A}$ is 4×5 (i.e., $\mathbf{A}$ has dimensions 4×5) and the vector $\mathbf{v}$ is 5×1, what are the dimensions of $\mathbf{Av}$ and $(\mathbf{Av})^T$?

Exercise 2.12

How do you find the transpose of a vector or matrix in MATLAB?

Matrix-Matrix Multiply

Matrices can be multiplied together to produce other matrices. In general, when you multiply a matrix $\mathbf{A}$ with another matrix $\mathbf{B}$, you need the matrix on the left side of the product to have the same number of columns as the number of rows in the matrix on the right side. In other words if $\mathbf{A}$ is $m \times n$, and $\mathbf{B}$ is $p \times q$, you need $n = p$ for the product $\mathbf{C} = \mathbf{AB}$ to be defined. The product results in a new matrix $\mathbf{C}$ which is $m \times q$. The q columns of the product matrix $\mathbf{C}$ are precisely the q vectors that would result from multiplying $\mathbf{A}$ with the vectors formed by the columns of $\mathbf{B}$.

Consider the following matrices

$$\mathbf{A} = \begin{bmatrix} 2 & 1 \\ 3 & -1 \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} 1 & 5 \\ -2 & 3 \end{bmatrix}.$$

The product of the two $\mathbf{C} = \mathbf{AB}$ is computed as follows

$$\mathbf{C} = \begin{bmatrix} 2 & 1 \\ 3 & -1 \end{bmatrix} \begin{bmatrix} 1 & 5 \\ -2 & 3 \end{bmatrix} = \begin{bmatrix} (2)(1) + (1)(-2) & (2)(5) + (1)(3) \\ (3)(1) + (-1)(-2) & (3)(5) + (-1)(3) \end{bmatrix} = \begin{bmatrix} 0 & 13 \\ 5 & 12 \end{bmatrix}$$

As a second example consider the matrices $\mathbf{A}$ and $\mathbf{B}$ defined below, and let the product $\mathbf{C} = \mathbf{AB}$.

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 2 \\ 4 & 1 \end{bmatrix} \mathbf{B} = \begin{bmatrix} 1 & 4 \\ 2 & 3 \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} (1)(1) + (2)(2) & (1)(4) + (2)(3) \\ (3)(1) + (2)(2) & (3)(4) + (2)(3) \\ (4)(1) + (1)(2) & (4)(4) + (1)(3) \end{bmatrix} = \begin{bmatrix} 5 & 10 \\ 7 & 18 \\ 6 & 19 \end{bmatrix}$$

As mentioned above, one way of envisioning matrix multiplication is if we consider the columns of input matrix $\mathbf{B}$ as a set of column vectors, we can multiply these column vectors one at a time by the matrix $\mathbf{A}$, and the resulting vectors will be the corresponding columns of the output matrix $\mathbf{C}$, i.e.

$$\mathbf{AB} = \mathbf{A}[\mathbf{B}_1, \mathbf{B}_2, \ldots] = [\mathbf{AB}_1, \mathbf{AB}_2, \ldots]$$

where $\mathbf{B}_1$ is the first column of matrix $\mathbf{B}$ etc.

Consider the following matrices:

$$\mathbf{A} = \begin{bmatrix} -2 & 4 \\ 0 & 3 \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 5 & -3 \\ -1 & -1 \end{bmatrix}$$

Exercise 2.13

Find the matrix product $\mathbf{AB}$.

Exercise 2.14

Find the matrix product $\mathbf{BA}$

Note that these two products are NOT equal. In general, matrix multiplication, unlike scalar multiplication, is NOT commutative. In other words, in general $\mathbf{AB} \neq \mathbf{BA}$. However, the distributive property IS valid for matrices: $\mathbf{A}(\mathbf{B} + \mathbf{C}) = \mathbf{AB} + \mathbf{AC}$ so long as we keep the order of the multiplication the same $(\mathbf{B} + \mathbf{C})\mathbf{A} = \mathbf{BA} + \mathbf{CA}$. Recall the definition of matrix addition: if two matrices are of the same size then they can be added and each entry of the new matrix is the sum of the entries of the original matrices, e.g.

$$\begin{bmatrix} 5 & -3 \\ -1 & -1 \end{bmatrix} + \begin{bmatrix} 4 & -2 \\ -3 & -1 \end{bmatrix} = \begin{bmatrix} 9 & -5 \\ -4 & -2 \end{bmatrix}$$

In addition to matrices $\mathbf{A}$ and $\mathbf{B}$ defined above, consider the matrix

$$\mathbf{C} = \begin{bmatrix} -5 & -1 \\ -3 & 2 \end{bmatrix}$$

Exercise 2.15

Calculate $\mathbf{A}(\mathbf{B} + \mathbf{C})$.

Exercise 2.16

Calculate $\mathbf{AB} + \mathbf{AC}$. Is it equal to your previous answer?

Finally, since matrix multiplication is defined, there is no reason not to multiply a matrix by itself. This only works if it is a square matrix. (Think about why this is true.) Using $\mathbf{A}$ and $\mathbf{B}$ from above, evaluate the following expressions

Exercise 2.17

1. $\mathbf{A}^2$
2. $\mathbf{B}^3$

Special Types of Matrices

Exercise 2.18

There are lots of matrices that are special. Use a trusted linear algebra reference to define the following types of matrices, and provide an example of each:

1. Square Matrix
2. Rectangular Matrix
3. Diagonal Matrix
4. Identity Matrix
5. Symmetric Matrix

2.4 Matrices as transformation operators

When matrices operate on (i.e., multiply) spatial position vectors, the vector which results is another spatial position vector. The original spatial position has been 'transformed' into another position. In particular,

there are specific matrices which accomplish specific desired transformations. These are used in many different disciplines.

Identity and Scaling Operations

The matrix

$$\mathbf{I} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.24}$$

when multiplying the vector

$$\mathbf{v} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2.25}$$

will reproduce the same vector, i.e. $\mathbf{I}\mathbf{v} = \mathbf{v}$. For this reason, the matrix $\mathbf{I}$ above is called an identity matrix. Identity matrices in higher dimensions are defined the same way, i.e., a 4-dimensional identity matrix is a 4×4 matrix with 1s on the diagonal and zeros everywhere else, i.e.,

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.26}$$

Exercise 2.19

1. Another important and simple operation is to be able to take a vector and scale (increase or decrease its length) it by an overall multiplicative factor while maintaining its direction. Consider the vector

   $$\mathbf{v} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

   Thinking about how the identity matrix acts on this vector, propose a 3×3 matrix which scales this vector by a factor of 3 to the vector

   $$3\mathbf{v} = \begin{bmatrix} 3x \\ 3y \\ 3z \end{bmatrix}.$$

   In other words, find a 3×3 matrix $\mathbf{M}$ such that $\mathbf{M}\mathbf{v} = 3\mathbf{v}$ for any vector $\mathbf{v}$.

2. What if you want to scale the x component differently than the y component? Write down the 3×3 matrix which scales the x component by 3 and the y component by 5 and leaves the z component the same.

3. Write down the 3×3 matrix which scales the x component by a, the y component by b, and the z component by c.

2.5 Data in Matrices and Vectors

Most of the examples you saw up to now in this assignment involved vectors which represent spatial positions, and most of the matrices you encountered represent transformations of the spatial vectors. But, as you saw with the example involving fruits, vectors can also be used to store data. So can matrices.

For instance, you may have the following matrix

$$\begin{bmatrix} 41 & 35 & 37 & 43 \\ 49 & 40 & 48 & 61 \end{bmatrix} \tag{2.27}$$

whose first row represents the forecasted high temperature in Needham for the next 4 days (as of the day this was written) and the second row represents the forecasted high temperatures for Washington DC. By representing this data in matrix form, you can do a number of operations to help extract useful information from the data.

Exercise 2.20

For this exercise, you will work with historical temperature data for the cities of Boston, Providence, Washington DC and New York.

1. Download the file `temps.mat` from canvas and load the data in it into MATLAB using `» load temps.mat`. You should now have access to a matrix `T` which contains daily average temperatures from 1995 to 2015 for the cities of Boston, Providence, Washington DC and New York (we are not telling you in what order yet). By using MATLAB's `size` function, determine the dimensions of this matrix. Are the temperatures for each city contained in the rows or the columns of this matrix?

2. The data provided is given in Fahrenheit, and suppose you wish to convert it to Celsius using matrix operations (note that there are a number of ways of doing this, but we are focusing on using matrices here). The formula for converting a Fahrenheit temperature to Celsius is to first subtract 32 from the Fahrenheit temperature, and multiply the result by $\frac{5}{9}$.

   a) Define a matrix of the same shape as `T` with all its entries equalling 32, and call this matrix B. You will find MATLAB's `ones` function, which generates a matrix filled with 1's, useful here.

   b) Define a square, diagonal matrix of the appropriate dimensions which when multiplying another matrix scales all its entries by $\frac{5}{9}$. You should call this matrix A. You will find MATLAB's `eye` function, which generates an identity matrix, useful here.

c) Using your answers to the previous parts and appropriate matrix operations, please provide 1 line of MATLAB code which generates a new matrix Y which contains the temperature data in Celsius.

3. Lets go back to Fahrenheit for the rest of the assignment. Extract the temperatures for each city into 4 different vectors `t1, t2, t3, t4`, and check that the dimensions of these vectors are as expected.

4. Using MATLAB's mean function, which computes the average of the values in a vector, and guess, based on geography, which of the vectors corresponds to the temperature for which city.

5. What are the maximum and minimum temperatures for Boston in the 20 years for which you have data?

6. On the same axes, plot graphs for the daily temperatures for the four cities for the last year for which you have data. Use MATLAB's `legend, xlabel, ylabel` functions to label the graphs.

7. Suppose that a genie told you that you can guess the temperature of New York, which we call T_n, using the temperatures of Boston, Providence, and Washington DC, which we respectively call T_b, T_p and T_w. From the matrix T, extract a 3×365 matrix of daily temperatures for the last year (for which you have data) in Boston, Providence and Washington DC.

8. The genie says that a good approximation for the temperature on a given day in New York is given by

$$T_n \approx 0.2235T_b + 0.4193T_p + 0.3856T_w. \tag{2.28}$$

Formulate a matrix equation which uses the matrix from the previous part and the formula from the genie to guess the daily temperature in New York for the last year. Apply this equation in MATLAB.

9. On the same axes, plot your prediction for the temperature in New York from the previous part, and the true temperature data which you extract from T. Is the prediction close?

In the course of this module, you will learn how to come up with the coefficients we provided here using historical data. (No, we don't actually have a genie.)

2.6 Conceptual Quiz

Please figure out the answer to these questions and mark your answer in Canvas. You can retake the quiz, as needed.

1. $\mathbf{A}$ is a 3×4 matrix and $\mathbf{B}$ is a 4×2 matrix. What is the size of $\mathbf{AB}$?

A. 2×3
B. 3×1
C. 3×2
D. The product is not defined.

2. What is the result of the following matrix product

$$\begin{bmatrix} 1 & 2 & -3 \\ 2 & -2 & 4 \\ 3 & -4 & 5 \end{bmatrix} \begin{bmatrix} 4 & 2 \\ -3 & -3 \\ 1 & 1 \end{bmatrix}$$

A.
$$\begin{bmatrix} -5 & -7 \\ 18 & 14 \\ 24 & 23 \end{bmatrix}$$

B.
$$\begin{bmatrix} -5 & -7 \\ 18 & 14 \\ 29 & 23 \end{bmatrix}$$

C.
$$\begin{bmatrix} -5 & 18 & 24 \\ -7 & 14 & 23 \end{bmatrix}$$

D.
$$\begin{bmatrix} -5 & -7 & 3 \\ 18 & 14 & 6 \\ 24 & 23 & 9 \end{bmatrix}$$

3. Match the following items (* means any number):

1. Rectangular Matrix — A. $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
2. Diagonal Matrix — B. $\begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix}$
3. Identity Matrix — C. $\begin{bmatrix} * & 0 & 0 \\ 0 & * & 0 \\ 0 & 0 & * \end{bmatrix}$
4. Symmetric Matrix — D. $\begin{bmatrix} * & * \\ * & * \\ * & * \end{bmatrix}$

4. Which of the following matrices will scale the length of any 2-D vector by $\frac{1}{2}$?

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

B.

$$\begin{bmatrix} \frac{1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{2}} \end{bmatrix}$$

C.

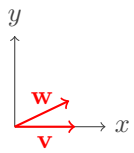
$$\begin{bmatrix} \frac{1}{2} & 0 \\ 0 & 1 \end{bmatrix}$$

D.

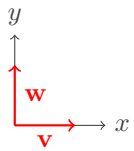
$$\begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

5. All of the following vectors are unit length. In which picture is $\mathbf{v} \cdot \mathbf{w}$ the largest?

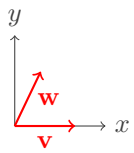
A.



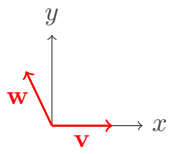
B.



C.



D.



Solution 2.1

1. Using the formula, $\mathbf{v} \cdot \mathbf{w} = \cos(\theta)$. So, when $\theta = 0$, (i.e., the vectors point in the same direction) the dot product is maximized and when $\theta = \pi/2$ (i.e., the vectors are perpendicular) the dot product is minimized.

2. The dot product is
$$\mathbf{v} \cdot \mathbf{w} = -2 + 0 - 4 + 18 = 12$$

Solution 2.2

1. Let $\mathbf{t} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$. Then $\mathbf{tf} = 1 + 2 + 3 = 6$, the total number of fruits in your refrigerator.

2. Let $\mathbf{c} = \begin{bmatrix} 1 & 1 & 0 \end{bmatrix}$. Then $\mathbf{cf} = 1 + 2 + 0 = 3$, the total number of citrus fruits in your refrigerator.

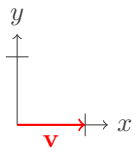
3. Let $\mathbf{w} = \begin{bmatrix} 120 & 250 & 100 \end{bmatrix}$. Then $\mathbf{wf} = 120 + 500 + 300 = 920$, the total weight of the fruits in your refrigerator.

Solution 2.3

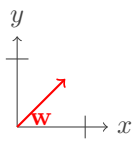
The vector $\mathbf{Gf}$ is a 2×1 vector whose first entry represents the total number of fruits and second entry represents the number of citrus fruits.

Solution 2.4

1. The vector $\mathbf{v}$ is



2. First, we compute
$$\mathbf{w} = \mathbf{A}\mathbf{v} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix},$$
which is visually represented as

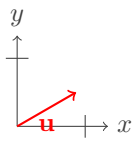


3. Multiplying $\mathbf{v}$ by $\mathbf{A}$ rotated the vector counterclockwise by 45 degrees.

4. First we compute

$$\mathbf{u} = \mathbf{B}\mathbf{v} = \begin{bmatrix} \frac{\sqrt{3}}{2} \\ \frac{1}{2} \end{bmatrix}$$

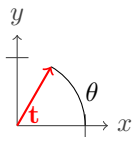
which is visually represented as



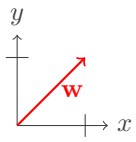
5. Multiplying $\mathbf{v}$ by $\mathbf{B}$ rotated the vector counterclockwise by 30 degrees.
6. First we compute

$$\mathbf{t} = \mathbf{R}\mathbf{v} = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}$$

which is visually represented as



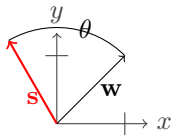
7. Multiplying $\mathbf{v}$ by $\mathbf{R}$ rotated the vector counterclockwise by θ degrees.
8. The vector $\mathbf{w}$ is



9. First we compute

$$\mathbf{s} = \mathbf{R}\mathbf{w} = \begin{bmatrix} \cos\theta - \sin\theta \\ \sin\theta + \cos\theta \end{bmatrix}$$

which is visually represented



10. Multiplying any vector by $\mathbf{R}$ rotates it by θ.

Solution 2.7

```
for n=1:4
M_squares(n,1) = n; %assigns the nth row, 1st column the value n
M_squares(n,2) = n^2; %assigns the nth row, 2nd column the value n^2
end
```

Solution 2.8

1. `[-3; -7; 4]`
2. `[-5 9]`
3. Does not work because the inner matrix dimensions must agree and here we have a 3×2 matrix multiplied by a 1×3 matrix
4. Does not work because the inner matrix dimensions must agree and here we have a 2×1 matrix multiplied by a 3×2 matrix
5. `[-3; -7]`
6. 9
7. Does not work because the index exceeds matrix dimensions. It is trying to access columns 2-4 of a two column matrix.
8. Does not work because the inner matrix dimensions must agree and here we have a 1×3 matrix multiplied by a 1×2 matrix.

Solution 2.9

$$4\mathbf{A} - 5\mathbf{B} = \begin{bmatrix} 7 & 6 & -11 \\ 2 & -6 & -1 \end{bmatrix} \tag{2.23}$$

Solution 2.10

The dimensions of $\mathbf{A}^T$ are 5×4.

Solution 2.11

$\mathbf{Av}$ is 4×1 and $(\mathbf{Av})^T$ is 1×4.

Solution 2.12

You use the apostrophe: $(\mathbf{A})^T$ is `A'` in Matlab.

Solution 2.13

$$\mathbf{AB} = \begin{bmatrix} -14 & 2 \\ -3 & -3 \end{bmatrix}$$

Solution 2.14

$$\mathbf{BA} = \begin{bmatrix} -10 & 11 \\ 2 & -7 \end{bmatrix}$$

Solution 2.15

$$\mathbf{A}(\mathbf{B} + \mathbf{C}) = \begin{bmatrix} -16 & 12 \\ -12 & 3 \end{bmatrix}$$

Solution 2.16

It is the same answer, as expected, since you can distribute matrices.

Solution 2.17

1. $$\mathbf{A}^2 = \begin{bmatrix} 4 & 4 \\ 0 & 9 \end{bmatrix}$$

2. $$B^3 = \begin{bmatrix} 152 & -72 \\ -24 & 8 \end{bmatrix}$$

Solution 2.18

1. A square matrix is one that has size $n \times n$, e.g.,
$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix}.$$
2. A rectangular matrix is one that has size $m \times n$ where n is not equal to m, e.g.,
$$\begin{bmatrix} * & * & * \\ * & * & * \end{bmatrix}.$$
3. A diagonal matrix is one whose only non-zero elements are on the diagonal from upper left to lower right, e.g.,
$$\begin{bmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 8 \end{bmatrix}.$$

4. The identity matrix is a square matrix with all zeroes except along the diagonal from the upper left to lower right, where the entries are all 1, e.g.,

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

5. A matrix is symmetric if it is square and equal to its own transpose, i.e. $A = A^T$, e.g.,

$$\begin{bmatrix} 1 & 7 & 3 \\ 7 & 4 & -5 \\ 3 & -5 & 6 \end{bmatrix}.$$

Solution 2.19

1.
$$\mathbf{M} = \begin{bmatrix} 3 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

2.
$$\mathbf{M} = \begin{bmatrix} 3 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3. We can generalize the result:
$$\mathbf{M} = \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{bmatrix}$$

Solution 2.20

1. After loading the temperatures you can see that they are stored inside a matrix called T which has 4 rows and 7670 columns, so presumably the temperature for each city is stored in a row.

2. a) We can define a matrix B of the same size as T but filled with 1s by typing » `B = ones(4,7670)` and then we can multiply it by 32 by typing » `B = 32*B`. Alternatively we could include the 32 from the start by typing » `B = 32*ones(4,7670)`

   b) There are two ways to do this: we can multiply either on the left or right.

   - Option 1: To multiply a 4×7670 matrix on the left, we need a 4×4 matrix, which we would create by typing » `A = eyes(4)*5/9`.
   - Option 2: To multiply a 4×7670 matrix on the left, we need a 7670×7670 matrix, which we would create by typing » `A = eyes(7670)*5/9`.

c) Now that we have the pieces in place we could simply type » `Y = A*(T-B)` (or `Y = (T-B)*A` if you chose the right multiply option). We can actually make this a lot simpler because MATLAB will create matrices of the correct size on the fly, so the following will work » `Y = (T-32)*5/9`. Normally we would expect `T-32` to be a problem because we are subtracting a scalar from a matrix, but MATLAB simply assumes that we wish to subtract 32 from every element in the matrix.

3. We can extract the first temperature by typing the following » `t1 = T(1,:)` - this simply grabs all of the elements in the first row, so that `t1` should be a row vector of size 1 by 7670. We create the other vectors in a similar way.

4. We can take the mean of the first city by typing » `mean(t1)` and we get 51.7667. The other means respectively are 51.9140, 58.4365, and 55.9451. A little bit of geography suggests that the cities are ordered as follows: Boston, Providence, DC, New York.

5. We can compute the maximum by typing » `max(t1)` and we get 90.7. The minimum is 0.7.

6. We are only supposed to grab the last year (365 days) so for Boston we would type » `plot(t1(end-364:end))`, or we could use the actual size of the vector.

7. Boston, Providence, and DC are stored in the first three rows. We'll extract their data and store it in a new matrix S by typing » `S = T(1:3,end-364:end)`, which grabs the first three rows and the last 365 entries.

8. We can define Tn by typing » `Tn = 0.2235*S(1,:) + 0.4193*S(2,:) + 0.3856*S(3,:)` since the city temperatures are stored in the each of the three rows of the matrix S.

9. Graphically they look pretty good. We can also examine the data a little more closely by looking at the difference between the predicted temperature and the actual temperature - it fluctuates with a mean of roughly 8.8115e-04, a maximum of 7.5334, and a minimum of -6.8966. Compared to the actual temperatures this implies that the prediction is never any worse than roughly 10%.

Chapter 3

Day 2: Matrix Transformations

3.1 Schedule

- 0900-0915: Debrief
- 0915-0945: Synthesis
- 0945-1030: 2D Rotations
- 1030-1045: Coffee
- 1045-1130: 3D Rotations
- 1130-1200: Reflections and Shearing
- 1200-1220: Review and Preview
- 1220-1230: Survey

3.2 Debrief

- With your table-mates, identify a list of key concepts/take home messages/things you learned in the assignment. Try to group them in categories like "Concepts", "Technical Details", "Matlab", etc.
- Try to resolve your confusions with your table-mates and by talking to an instructor.

3.3 Synthesis

Exercise 3.1

These are fundamental ideas about matrices and it is important to complete these. They should be done by hand.

1. What is the difference between a scalar, a vector, a matrix, and an array?
2. What are the rules for adding matrices?
3. When can two matrices be multiplied, and what is the size of the output?
4. What is the distributive property for matrix multiplication?
5. What is the associative property for matrix multiplication?
6. What is the commutative property for matrix multiplication?

Exercise 3.2

These are synthesis problems. It would be helpful to complete these. They should be done by hand.

1. Consider the matrix $\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$. Show that $\mathbf{A}^2$ commutes with $\mathbf{A}$.
2. Use the distribution law to expand $(\mathbf{A} + \mathbf{B})^2$ assuming that $\mathbf{A}$ and $\mathbf{B}$ are matrices of appropriate size. How does this compare to the situation for real numbers?
3. Show that $\mathbf{D} = \begin{bmatrix} 4 & -2 \\ 3 & -3 \end{bmatrix}$ satisfies the matrix equation $\mathbf{D}^2 - \mathbf{D} - 6\mathbf{I} = \mathbf{0}$.

Exercise 3.3

These are challenge problems. Pick one of them to wrestle with. It is not important to complete these. They should be done by hand.

1. The matrix exponential is defined by the power series

$$\exp \mathbf{A} = \sum_{k=0}^{\infty} \frac{\mathbf{A}^k}{k!}$$

   Assume $\mathbf{A} = \begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}$. Find a formula for $\exp \mathbf{A}$.
2. The real number 0 has just one square root: 0. Show, however, that the 2×2 zero matrix has infinitely many square roots by finding all 2×2 matrices $\mathbf{A}$ such that $\mathbf{A}^2 = \mathbf{0}$.
3. Use induction to prove that $\mathbf{A}^n$ commutes with $\mathbf{A}$ for any square matrix $\mathbf{A}$ and positive integer n.

3.4 2D Rotation Matrices

We're going to think about how to use rotation matrices to rotate a geometrical object. In doing so we will solidify fundamental concepts around matrix multiplication and start to explore the notion of "inverse". For

clarity we will first work in 2D. Recall that the rotation matrix $\mathbf{R}(\theta)$:

$$\mathbf{R}(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

will rotate an object counterclockwise **about the origin** through an angle of θ.

Exercise 3.4

This is a hands-on, conceptual problem involving the multiplication of 2D rotation matrices.

1. Place an object on your table, and imagine that the origin of an xy-coordinate system is at the center of your object with $+z$ pointing upwards.
2. Rotate it counterclockwise by 30 degrees, and then again by another 60 degrees. What is it's orientation now? How would you get there in one rotation instead? What does this suggest about the multiplication of rotation matrices?
3. What happens if you first rotate it by 60 degrees, and then by 30 degrees? What does this suggest about the commutative property of 2D rotation matrices?

Exercise 3.5

This is an algebra problem involving the multiplication of 2D rotation matrices.

1. Use some algebra to show that 2D rotation matrices commute, i.e. $\mathbf{R}(\theta_1)\mathbf{R}(\theta_2) = \mathbf{R}(\theta_2)\mathbf{R}(\theta_1)$.
2. Use some algebra to show that $\mathbf{R}(\theta_1)\mathbf{R}(\theta_2) = \mathbf{R}(\theta_1 + \theta_2)$. You will need to look up some trig identities.

Exercise 3.6

Now, consider a rectangle of width 2 and height 4, centered at the origin. For clarity, this means that the corners of the rectangle have coordinates $(1, 2)$, $(-1, 2)$, $(-1, -2)$, and $(1, -2)$.

1. Plot these four points by hand and connect them with lines to complete the rectangle.
2. Now, using the appropriate rotation matrix, transform each of the corner points by a rotation through 30 degrees counterclockwise (recall that the sin and cos of 30 degrees can be expressed

exactly). Compute and plot the resulting points by hand and connect them with lines. Does the resulting figure look like you'd expect?

Exercise 3.7

Now, let's do it in MATLAB.

1. Create and plot the original 4 points:$(1, 2)$, $(-1, 2)$, $(-1, -2)$, and $(1, -2)$. Then create the matrix that rotates them by 30 degrees counterclockwise, transform each of the four original points using the rotation matrix, and plot the resulting points. Does this look right? *Reminder:* `plot(1,2,'x')` *puts a mark at the point (1,2). Matlab: the functions cos and sin expect radians, while cosd and sind expect degrees.*
2. Operating on individual points with the rotation matrix is cool, but we can be much more efficient by operating on all 4 points at the same time. Write down the matrix whose columns represent the four corners of the rectangle. Then write down the matrix multiplication problem we can solve to transform the rectangle from above all at once. Create these matrices in MATLAB to perform the rotation in a single operation. Plot the resulting matrix to confirm your transformation! *Some MATLAB tips:* `plot(X,Y)` *creates a line plot of the values in the vector Y versus those in the vector X. So if you wanted to plot a line from the origin (0,0) to the point (1,2), you would do this: plot([0 1],[0 2]).* The command `axis([-xlim xlim -ylim ylim])` sets the axes of the current plot to run from `-xlim` to `xlim` and from `-ylim` to `ylim`
3. What is the area of the rectangle before and after the rotation?
4. What matrix should you use to undo this rotation? Define it in MATLAB and check.
5. Show on the board that the product of this matrix with the original rotation matrix is the identity matrix. For clarity, let's give this matrix the symbol $\mathbf{R}^{-1}$. It is the matrix that inverts the original operation and is known as the *inverse* of the matrix $\mathbf{R}$.

3.5 3D Rotations

We can extend the idea of 2D rotations to 3D rotations. The simplest approach is to think of 3D rotations as a composition of rotations about different axes. First let's define the rotation matrices for counterclockwise

rotations of angle θ about the x, y and z axes respectively.

$$\mathbf{R}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \tag{3.1}$$

$$\mathbf{R}_y = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{3.2}$$

$$\mathbf{R}_z = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

For example, to first rotate a vector $\mathbf{v}$ counterclockwise by θ about the x axis followed by counterclockwise by ϕ about the z axis, you need to do the following

$$\begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \mathbf{v} \tag{3.4}$$

We will next look at some sequence of physical rotations and relate them to these rotation matrices.

Exercise 3.8

Hold a closed book in front of you, with the top of the book towards the ceiling ($+z = (0, 0, 1)$ direction) and the cover of the book pointed towards you ($+x = (1, 0, 0)$ direction), which leaves the opening side of the book pointing towards your right ($+y = (0, 1, 0)$) and the spine toward the left.

1. Rotate the book by 90 degrees counter-clockwise about the x-axis, then from this position, rotate the book by 90 degrees counter-clockwise about the z-axis. Which direction is the cover of the book facing now?
2. Return to the starting position. Now rotate the book by 90 degrees counter-clockwise about the z axis, and then from this position, rotate the book by 90 degrees counter-clockwise about the x axis. Which direction is the cover of the book facing now? Is it the same as in part a?
3. An operation "commutes" if changing the order of operation doesn't change the result. Do 3D rotations commute?
4. The cover of the book is originally pointed towards $(1, 0, 0)$. Multiply this vector with the appropriate sequence of rotation matrices from above to reproduce your motions from part 1. Do you end up with the correct final cover direction?
5. Multiply the $(1, 0, 0)$ vector with the appropriate sequence of rotation matrices to reproduce the motions from part 2. Do you end up with the correct final cover direction?

6. Multiply the result of the previous part by the appropriate sequence of rotation matrices to return to the original $(1, 0, 0)$ vector.
7. From either of your answers to part 4 or part 5, try, instead of operating on the $(1, 0, 0)$ vector sequentially with one rotation matrix and then the other, take the product of the two rotation matrices first, and then multiply $(1, 0, 0)$ with the resultant matrix. Does this reproduce your answer?
8. Based on your answers to the previous parts, show that $(\mathbf{R}_z\mathbf{R}_x)^{-1} = \mathbf{R}_x^{-1}\mathbf{R}_z^{-1}$. This is a general property of matrix inverses – it works for all square, invertible matrices, not just rotation matrices!

3.6 Reflection and Shearing

In this activity we will meet reflection and shearing matrices, which will allow us to explore transformation matrices in general.

Reflection

Exercise 3.9

What do the following *reflection* matrices do? Think about it first, draw some sketches and then test your hypothesis in MATLAB using the rectangle with vertices $(0, 0)$, $(2, 0)$, $(2, 1)$, and $(0, 1)$. How much does the area of your basic rectangle change, if at all? What is the inverse of each?

1. $$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$
2. $$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$
3. $$\begin{bmatrix} \cos 2\theta & \sin 2\theta \\ \sin 2\theta & -\cos 2\theta \end{bmatrix}$$

Shearing

Exercise 3.10

What do the following *shearing* matrices do? Think about it first, draw some sketches and then test your hypothesis in MATLAB with the rectangle with vertices $(0,0)$, $(2,0)$, $(2,1)$, and $(0,1)$. How much does the area of your basic rectangle change, if at all? What is the inverse of each?

1. $$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$
2. $$\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$
3. $$\begin{bmatrix} 1 & 2k \\ 0 & 1 \end{bmatrix}$$
4. $$\begin{bmatrix} 1 & 0 \\ 2k & 1 \end{bmatrix}$$

Review and Preview

Solution 3.1

1. Scalars, vectors, and matrices are examples of arrays. A 0-dimensional array can be thought of as a scalar. A 1-dimensional array is a vector. A 2-dimensional array is a matrix.
2. The matrices have to be the same size and addition is element-wise.
3. The matrices have to compatible (inner dimensions agree), and the output is dictated by the outer dimensions, i.e. $(n \times m)(r \times s) = (n \times s)$.
4. Distributive property: $\mathbf{A}(\mathbf{B} + \mathbf{C}) = \mathbf{AB} + \mathbf{AC}$
5. Associative property: $\mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C}$
6. Commutative property: Two matrices commute if $\mathbf{AB} = \mathbf{BA}$ but this is not always true.

Solution 3.2

1. You need to show that $\mathbf{A}^2\mathbf{A} = \mathbf{A}\mathbf{A}^2$ for this particular matrix. You can do it by multiplying.
2. Using the distributive property you can see that $(\mathbf{A} + \mathbf{B})^2 = (\mathbf{A} + \mathbf{B})(\mathbf{A} + \mathbf{B}) = \mathbf{A}^2 + \mathbf{AB} + \mathbf{BA} + \mathbf{B}^2$
3. If you plug $\mathbf{D}$ and $\mathbf{D}^2$ into the equation you should find that the result is a zero matrix.

Solution 3.3

1. The matrix exponential is defined by the power series $\exp \mathbf{A} = \mathbf{I} + \mathbf{A} + \frac{\mathbf{A}^2}{2!} + \ldots$. Notice that this $\mathbf{A}$ is diagonal and $\mathbf{A}^2 = \begin{bmatrix} 2^2 & 0 \\ 0 & 3^2 \end{bmatrix}$ and the exponential becomes $\exp \mathbf{A} = \begin{bmatrix} 1 + 2 + 2^2/2! + \ldots & 0 \\ 0 & 1 + 3 + 3^2/2! + \ldots \end{bmatrix}$. If you have seen power series before then you will recognise that $\exp \mathbf{A} = \begin{bmatrix} \exp 2 & 0 \\ 0 & \exp 3 \end{bmatrix}$.
2. You can define a general two by two matrix $\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$, find $\mathbf{A}^2$, set each of the entries equal to zero and find constraints on the entries a, b, c, d.
3. You need to show that $\mathbf{A}^n\mathbf{A} = \mathbf{A}\mathbf{A}^n$ for any square matrix $\mathbf{A}$ and any positive integer n by induction. First you show it is true for $n = 1$ and $n = 2$. Then assume it is true for some $n = k$, and prove that it must be true for $n = k + 1$. You use the fact that $\mathbf{A}$ commutes with itself and the associative property, i.e $\mathbf{A}^2\mathbf{A} = (\mathbf{AA})\mathbf{A} = \mathbf{A}(\mathbf{AA}) = \mathbf{A}\mathbf{A}^2$.

Solution 3.4

1. Okay, I placed my book on the table.

2. You could get there by rotating once by 90 degrees. This suggests that the product of two rotation matrices of angles θ_1 and θ_2 is a rotation matrix of $\theta_1 + \theta_2$, i.e. $\mathbf{R}(\theta_1)\mathbf{R}(\theta_2) = \mathbf{R}(\theta_1 + \theta_2)$.

3. You end up in the same orientation so it doesn't matter the order. This suggests that the order of multiplication doesn't matter so that two rotation matrices must commute.

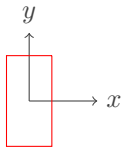
Solution 3.5

1. You could multiply out two rotation matrices with angle θ_1 and θ_2 in the two different orders and you will observe that the output is the same because real numbers commute, i.e. $\cos\theta_1 \cos\theta_2 = \cos\theta_2 \cos\theta_1$.

2. If you multiply two matrices together you will get the following expression in the first row and first column, $\cos\theta_1 \cos\theta_2 - \sin\theta_1 \sin\theta_2$. You will find a trig identity which reduces this to $\cos(\theta_1 + \theta_2)$. Similar reductions take place for the other elements.

Solution 3.6

1. The rectangle is



2. The rotation matrix is

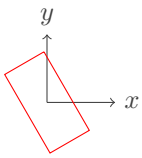
$$\mathbf{R} = \begin{bmatrix} \cos 30 & -\sin 30 \\ \sin 30 & \cos 30 \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{3}}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{2} \end{bmatrix}.$$

Applying this to each point, we get

$$\mathbf{R}\begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{3}-2}{2} \\ \frac{1+2\sqrt{3}}{2} \end{bmatrix}, \quad \mathbf{R}\begin{bmatrix} 1 \\ -2 \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{3}+2}{2} \\ \frac{1-2\sqrt{3}}{2} \end{bmatrix},$$

$$\mathbf{R}\begin{bmatrix} -1 \\ 2 \end{bmatrix} = \begin{bmatrix} \frac{-\sqrt{3}-2}{2} \\ \frac{-1+\sqrt{3}}{2} \end{bmatrix}, \quad \mathbf{R}\begin{bmatrix} -1 \\ -2 \end{bmatrix} = \begin{bmatrix} \frac{-\sqrt{3}+2}{2} \\ \frac{-1-\sqrt{3}}{2} \end{bmatrix}.$$

And the rotated figure looks like,



Solution 3.7

1. There are lots of ways to do this point by point. Here is an example of how to transform the bottom right point:

```
>> BR = [1;-2]
>> plot(BR(1,:),BR(2,:),'b*')
>> rotmatrix = [cosd(30) -sind(30); sind(30) cosd(30)]
>> nBR = rotmatrix*BR
>> plot(nBR(1,:),nBR(2,:),'r*')
```

2. There are lots of ways to do this. Here is an example where we include the first point twice so that the points can easily be connected with lines:

```
>> pts = [1 -1 -1 1 1;2 2 -2 -2 2]
>> npts = rotmatrix*pts
>> plot(pts(1,:),pts(2,:),'b'), hold on
>> plot(pts(1,:),pts(2,:),'r')
>> axis([-3 3 -3 3])
>> axis equal
```

3. The area of the rectangle is the same before and after rotation: 8 square units.

4. To undo this rotation you could simply rotate it by 30 degrees clockwise, using the matrix

$$\mathbf{R}^{-1} = \begin{bmatrix} \cos 30 & \sin 30 \\ -\sin 30 & \cos 30 \end{bmatrix}.$$

5. The product of $\mathbf{R}^{-1}$ and $\mathbf{R}$ is

$$\mathbf{R}^{-1}\mathbf{R} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} = \begin{bmatrix} \cos^2\theta + \sin^2\theta & 0 \\ 0 & \cos^2\theta + \sin^2\theta \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

where we have used the trig identity $\cos^2\theta + \sin^2\theta = 1$.

Solution 3.8

1. The cover is now facing toward the $+y$ axis (the positive part of the y axis).

2. The cover is now facing the $+z$ axis. This is different than in part a.

3. Since the answers for the first two parts are different, 3D rotations do not commute.

4. Let $\mathbf{v}$ be the vector that represents the initial direction of the cover of the book,

$$\mathbf{v} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}.$$

Rotation by 90 degrees counterclockwise around the x axis is given by

$$\mathbf{R}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}.$$

so that the new vector becomes

$$\mathbf{R}_x\mathbf{v} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

Rotation by 90 degrees counterclockwise around the z axis is given by

$$\mathbf{R}_z = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

so that the new vector becomes

$$\mathbf{R}_z\mathbf{R}_x\mathbf{v} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

which is the correct final direction.

5. Using the matrices from above,

$$\mathbf{R}_x\mathbf{R}_z\mathbf{v} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

6. To rotate 90 degrees clockwise around the x axis we use the matrix

$$\mathbf{R}_x^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$$

and to rotate 90 degrees clockwise around the z axis we use the matrix

$$\mathbf{R}_z^{-1} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Then we can return the vector $(0, 0, 1)$ to its original position $(1, 0, 0)$ by

$$\mathbf{R}_z^{-1}\mathbf{R}_x^{-1}\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}.$$

7. We can multiply the rotation matrices together and perform a single matrix multiplication. For part d, the relevant matrix product is

$$\mathbf{R}_z\mathbf{R}_x = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

and we see that

$$\mathbf{R}_z\mathbf{R}_x \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

as expected.

8. We can see from the previous parts that

$$(\mathbf{R}_z\mathbf{R}_x)^{-1} = \mathbf{R}_x^{-1}\mathbf{R}_z^{-1}.$$

In other words, when you take the inverse, the order of operations must swap!

Solution 3.9

1. This matrix reflects everything over the y-axis. In the figure below, the original blue rectangle becomes the orange rectangle. The area of the rectangle stays the same.

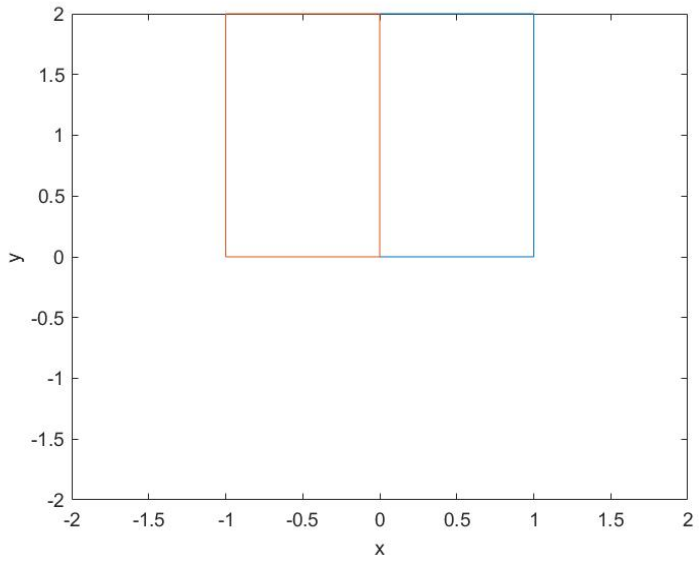


Figure 3.1: Reflection over y-axis.

2. This matrix reflects everything over the x-axis. In the figure below, the original blue rectangle becomes the orange rectangle. The area of the rectangle stays the same.

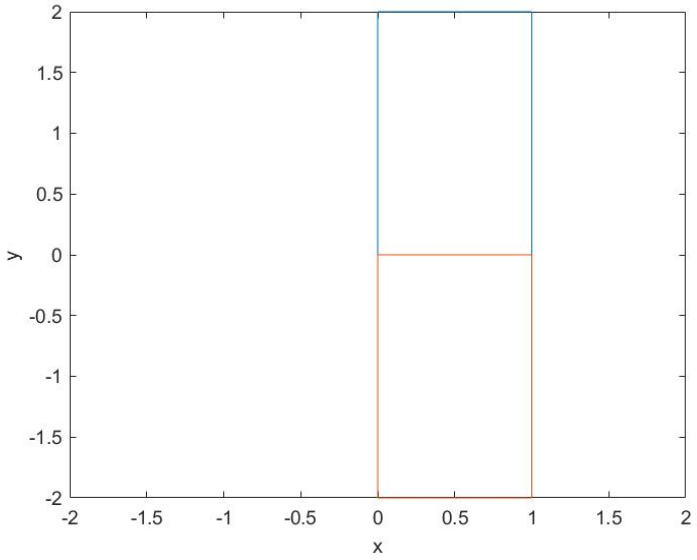


Figure 3.2: Reflection over x-axis.

3. For example, let $\theta = 30$ degrees. Then the rectangle is reflected along the line that is 30 degrees counterclockwise from the x-axis. In the figure below, the original blue rectangle becomes the orange rectangle.

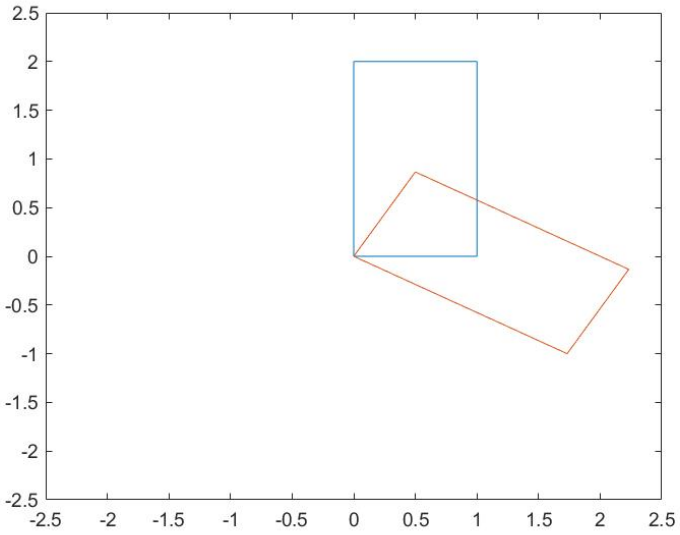


Figure 3.3: Reflection over 30 degree line.

Notice that, if we plug in $\theta = 90$, we get the matrix from part 1, which reflects over the x-axis (i.e., 90 degree line) and, if we plug in $\theta = 0$, we get the matrix from part 2, which reflects over the y-axis (i.e., the 0 degree line).

Solution 3.10

1. This shearing matrix pulls the points along horizontal lines and the strength of the pull is proportional to the y coordinate. In the figure below, the blue rectangle is sheared to become the orange rectangle:

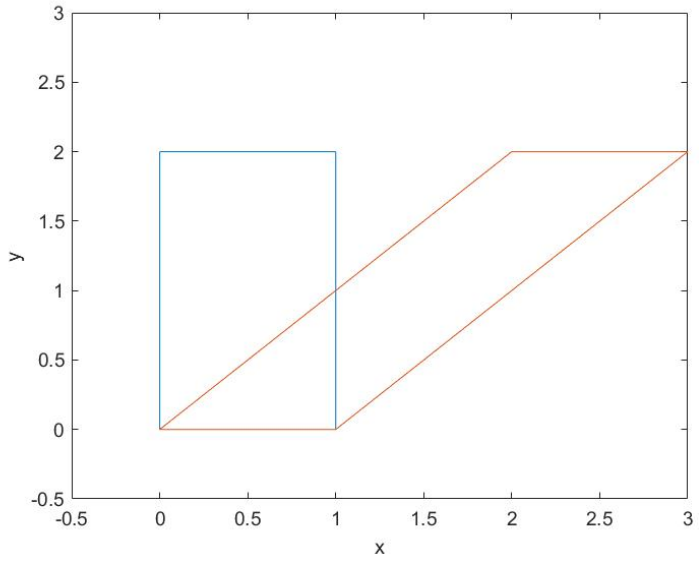


Figure 3.4: Shearing in x direction.

The area of the rectangle does not change. The inverse is

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

2. This shearing matrix pulls the points along vertical lines and the strength of the pull is proportional to the x coordinate. In the figure below, the blue rectangle is sheared to become the orange rectangle:

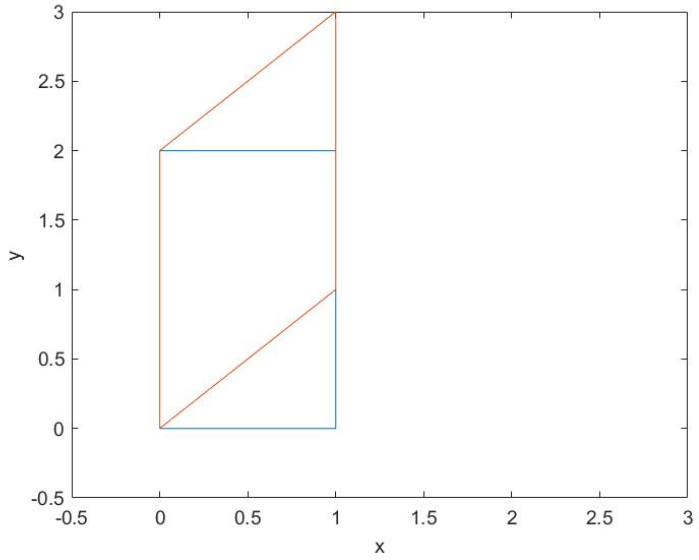


Figure 3.5: Shearing in y direction.

The area of the rectangle does not change. The inverse is

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

3. This shearing matrix pulls the points along horizontal lines and the strength of the pull is proportional to the y coordinate and the constant k (the bigger the k, the stronger the pull). In the figure below, with $k = 2$, the blue rectangle is sheared to become the orange rectangle:

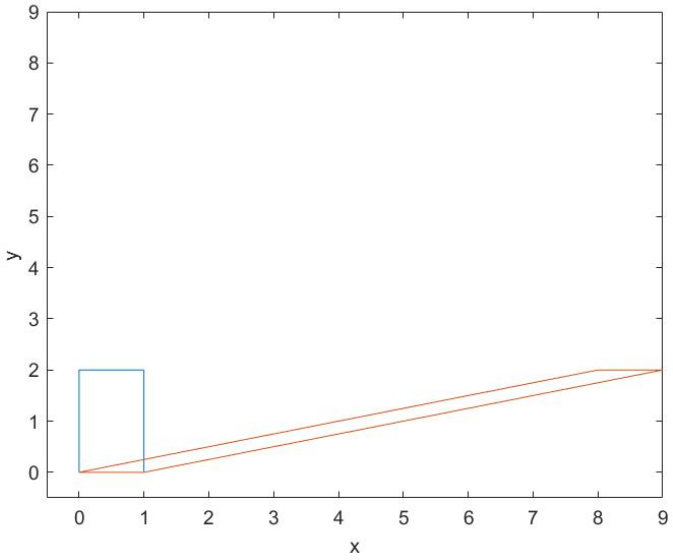


Figure 3.6: Shearing in x direction with $k = 2$.

The area of the rectangle does not change. The inverse is

$$\begin{bmatrix} 1 & -4 \\ 0 & 1 \end{bmatrix}.$$

4. This shearing matrix pulls the points along vertical lines and the strength of the pull is proportional to the x coordinate and the constant k (the bigger the k, the stronger the pull). In the figure below, with $k = 2$, the blue rectangle is sheared to become the orange rectangle:

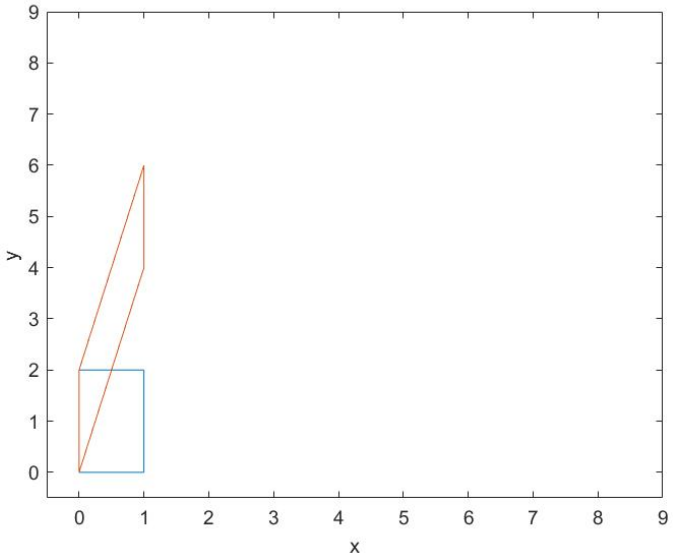


Figure 3.7: Shearing in y direction with $k = 2$.

The area of the rectangle does not change. The inverse is

$$\begin{bmatrix} 1 & 0 \\ -4 & 1 \end{bmatrix}.$$

Chapter 4

Night 2: Matrix Operations

Overview and Orientation

Learning Objectives

Concepts

- Compute the determinant of a 2×2 matrix
- Know the relationship between the determinant of a matrix and whether the matrix is invertible
- Find the inverse of a 2×2 matrix by hand
- Use computational tools to find the inverse of an $n \times n$ matrix
- Design a 2 or 3-dimensional matrix that will scale a vector by given amounts in the x, y or z direction
- Design a 3-dimensional matrix that will translate a 2-D vector by given amounts in x and y

MATLAB skills

- Represent a set of points in 2-D space (i.e., pairs of x, y values) as column vectors
- Transform a set of 2-D points (i.e., the outline of a shape) using a matrix to rotate and translate the original
- Multiply matrices and find their inverses
- Compute the determinant of a matrix

Suggested Approach

See Night 1 assignment for our general suggested approach to night assignments and a list of linear algebra resources.

4.1 Determinant of a Matrix

The determinant of a square matrix is a property of the matrix which indicates many important things, including whether a matrix is invertible or not. We will see more of this when we see matrix inverses shortly. The determinant of a matrix $\mathbf{G}$ is denoted a few different ways.

$$\det(\mathbf{G}) = |\mathbf{G}| \tag{4.1}$$

Consider a generic 2×2 matrix $\mathbf{G}$:

$$\mathbf{G} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

The formula for the determinant of a 2×2 matrix is quite straightforward:

$$\det(\mathbf{G}) = ad - bc \tag{4.2}$$

For example, for the following 2×2 matrix,

$$\begin{aligned} \det\left(\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}\right) &= \begin{vmatrix} 1 & 2 \\ 3 & 4 \end{vmatrix} \\ &= (1)(4) - (2)(3) = -2 \end{aligned} \tag{4.3}$$

Exercise 4.1

Return to the transformation matrices in the day assignment and calculate the determinant for the following:

1. The generic 2×2 rotation matrix

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

2. The matrix which reflects over the y axis

$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

3. The matrix which shears in the horizontal direction

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

Exercise 4.2

1. What do the following matrices do? Think about it first, draw some sketches and then test your hypothesis in MATLAB. How much does the area of your basic rectangle change, if at all?

a)

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

b)

$$\begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}$$

2. Is it possible to "undo" the matrices above? Why or why not?

Exercise 4.3

1. What are the determinants of the two matrices from the previous exercise, Exercise 4.2?
2. Generalizing from Exercise 4.1 and Exercise 4.2, what's the relationship between the determinant of a matrix and the result of transforming a rectangle by that matrix?

Finding the determinant of an $n \times n$ matrix, where $n > 2$, is a bit more computationally intensive. If you want to learn how to do the procedure by hand, check out this Khan Academy video. For this course, we simply recommend you use the `det` function in MATLAB.

4.2 Matrix Inverses

Inverse of 2×2 *Matrices*

In class you worked with rotation matrices and transformations that were compositions of simpler rotations, and you learned how to invert them. When you multiply a vector by any matrix (not just ones that are associated with simple spatial transformations), you transform the original vector into a new vector. More generally (than rotations), you can *often* undo the linear transformation (just like you did with the rotation matrix). Undoing this linear transformation is a linear transformation itself! Therefore the act of undoing a linear transformation can be formulated with a matrix multiply.

Exercise 4.4

Consider the following matrices and vector. (Don't try to interpret these as intuitive geometrical operations; we're just using them to explore the determinant.) Work out the following problems in

MATLAB.

$$\mathbf{P} = \begin{bmatrix} 2 & 1 \\ 4 & 3 \end{bmatrix} \tag{4.4}$$

$$\mathbf{Q} = \begin{bmatrix} \frac{3}{2} & -\frac{1}{2} \\ -2 & 1 \end{bmatrix} \tag{4.5}$$

$$\mathbf{u} = \begin{bmatrix} 2 \\ 3 \end{bmatrix} \tag{4.6}$$

1. Find $\mathbf{w} = \mathbf{P}\mathbf{u}$.
2. Find $\mathbf{Q}\mathbf{w}$. How is this related to $\mathbf{u}$?
3. Find $\mathbf{Q}\mathbf{P}$. Does the answer look familiar?
4. Find $\mathbf{P}\mathbf{Q}$.
5. Find the determinant of $\mathbf{P}$. In MATLAB, you can compute the determinant of any (not just 2×2) matrix using the `det` function.
6. Find the determinant of $\mathbf{Q}$.

A matrix $\mathbf{B}$ is said to be the inverse of the matrix $\mathbf{A}$ if, and only if, $\mathbf{B}\mathbf{A} = \mathbf{I}$ and $\mathbf{A}\mathbf{B} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. For 2×2 matrices, the inverse (if it exists) is given by the following

$$\mathbf{G} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \tag{4.7}$$

$$\mathbf{G}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} \tag{4.8}$$

The last equation should indicate to you that the inverse of the matrix $\mathbf{G}^{-1}$ is only defined if $ad - bc \neq 0$. Sweet mother of linear algebra, $ad - bc$ is our buddy the determinant. More generally, any square matrix can be inverted if and only if its determinant is non-zero.

Now let's practice calculating inverses, some of their properties, and how we may use them.

Exercise 4.5

All matrices $\mathbf{A}$ and $\mathbf{B}$ which have inverses have the following properties

$$(\mathbf{A}\mathbf{B})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}$$
$$(\mathbf{A}^T)^{-1} = (\mathbf{A}^{-1})^T$$

1. Using the above properties, please compute the following by hand.

a) If

$$\mathbf{P} = \begin{bmatrix} 2 & 1 \\ 4 & 3 \end{bmatrix} \tag{4.9}$$

$$\mathbf{B} = \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix} \tag{4.10}$$

$$\tag{4.11}$$

find $(\mathbf{PB})^{-1}$. Recall that you already know the inverse of $\mathbf{P}$ from earlier.

b) For $\mathbf{P}$ as defined above, find

$$(\mathbf{P}^T)^{-1} \tag{4.12}$$

2. Use the inverse formula to calculate the inverses for the first three matrices in Exercise 4.1. Confirm your answers by multiplying the inverse with the original matrix.

a) By hand, write an equation relating $\mathbf{n}$ and $\mathbf{d}$, using a matrix-vector product.

b) By hand, calculate how many oranges and apples you have.

c) Why do you think this type of problem is often called an inverse problem?

Note that solving matrix-vector equations like above can be done without explicitly computing the matrix inverse which is computationally expensive. (A nod to our future friend, left matrix divide or backslash divide.)

Inverse of $n \times n$ Matrices

For higher-dimensional matrices, e.g. $n \times n$ matrices for $n > 2$, the matrix inverse is defined in the same way. Suppose you have an $n \times n$ matrix $\mathbf{A}$ and an $n \times n$ matrix $\mathbf{B}$. Then $\mathbf{B}$ is the inverse of $\mathbf{A}$ if and only if $\mathbf{BA} = \mathbf{I}$ and $\mathbf{AB} = \mathbf{I}$. The following are some properties of inverses of matrices

- Only square matrices are invertible, i.e., only square matrices have inverses.
- A matrix has an inverse only if its determinant is non-zero.

There are a number of different procedures to compute the inverse of higher-dimensional matrices, but we will not be going into the details of their computation here. You can look them up if you are interested, or need to in the future. In MATLAB, you can compute the inverse of a matrix using the `inv` function.

Exercise 4.6

1. Consider the example with the fruits that you worked out earlier. Now, in addition to apples

and oranges, suppose you also had an unknown number of pears which each weigh 3 oz, and cost \$3. Additionally, suppose that the total weight of the fruits is 45 oz, and you paid a total of \$21 for the fruit.

a) If possible find the numbers of oranges, apples and pears. If not, please explain why.

b) Suppose that you additionally know that you have a total of 14 fruits. Can you formulate and solve a matrix-vector equation to find out the numbers of oranges, apples and pears you have?

c) What is the determinant of the matrix you have set up to solve this?

2. The fruit vendors bought the pricing algorithm from Uber. Oranges are still \$2, pears are now only \$1.50, and (due to an influx of teachers) apples are now surging at \$1.50 each. Their weights stay the same. You return to the market, and again purchase 14 fruits, which have the same total weight and total cost.

a) Can you formulate and solve a matrix-vector equation to find out the numbers of oranges, apples and pears you have?

b) What is the determinant of the matrix you have set up to solve this?

c) Debrief at your table about what this means.

4.3 Transformation Matrices, Continued

Scaling

Returning to two dimensions. In the Night 1 assignment, you also learned about scaling matrices. Recall that the scaling matrix $\mathbf{S}$ scales the x-component by s_1 and the y-component by s_2

$$\mathbf{S} = \begin{bmatrix} s_1 & 0 \\ 0 & s_2 \end{bmatrix}.$$

Let's assume for the moment that $s_1 = 2$ and $s_2 = 1/3$. Working with the rectangles defined in class whose corners have coordinates $(1, 2)$, $(1, -2)$, $(-1, 2)$, and $(-1, -2)$ complete the following activities:

Exercise 4.7

1. Predict what would happen if you operate on the rectangle with $\mathbf{S}$.
2. Write a MATLAB script to carry out this operation and check your prediction.
3. How does the area of the rectangle change?

4. What matrix should you use to *undo* this scaling? Show that the product of this matrix with the original scaling matrix is the *identity* matrix.
5. Define it in MATLAB and check. Again, this is the *inverse* matrix and we give it the symbol $\mathbf{S}^{-1}$.
6. In MATLAB, change the value of s_2 to 1 and find the product of the new $\mathbf{S}$ and your rectangle. How does the area of the rectangle change? Change the value of s_2 back to 1/3.
7. Predict what would happen if you operate on the original rectangle with $\mathbf{SR}$, where $\mathbf{R}$ is the rotation matrix. How about $\mathbf{RS}$? Implement both of these in MATLAB and check.
8. How would you *undo* each of these operations ($\mathbf{SR}$ and $\mathbf{RS}$)? How is the inverse of the product related to the individual inverses, i.e. what is the relationship between $(\mathbf{SR})^{-1}$ and $\mathbf{S}^{-1}$ and $\mathbf{R}^{-1}$? What about $(\mathbf{RS})^{-1}$?

Translation

It would be really useful if, in addition to scaling and rotating our objects, we could translate them. Let's start by thinking about vectors and then we will figure out how to represent translation as a matrix operation.

Consider an initial vector $\mathbf{v}$ and a translation vector $\mathbf{t}$. The new translated vector is simply $\mathbf{v} + \mathbf{t}$. For example, if you start with the initial vector $\mathbf{v} = \begin{bmatrix} x \\ y \end{bmatrix}$ and translate it using the vector $\begin{bmatrix} 2 \\ 3 \end{bmatrix}$ then the new vector is just $\begin{bmatrix} x+2 \\ y+3 \end{bmatrix}$. More generally, if the translation vector is $\begin{bmatrix} t_x \\ t_y \end{bmatrix}$ then the new vector will be $\begin{bmatrix} x+t_x \\ y+t_y \end{bmatrix}$.

Wouldn't it be handy if we could define translation as a matrix operation? Yes, indeed it would be, we hear you say. Here is the standard method: add another entry to the original vector, and set it equal to 1, i.e., $\mathbf{v} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$. Now define the translation matrix as

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}.$$

Exercise 4.8

1. Show that $\mathbf{Tv}$ accomplishes the process of translation (if you ignore the third entry in the new vector). What is the final vector?
2. Predict what would happen if you operate on our old friend the rectangle with the translation

matrix defined by $t_x = 2$ and $t_y = 3$.

3. Write a MATLAB script to carry out this operation and check your prediction. How has the area of your rectangle changed?
4. What matrix should you use to *undo* this translation? Show on paper that the product of this matrix with the original translation matrix is the *identity* matrix. Define it in MATLAB and check. Again, this is the *inverse* matrix and we give it the symbol $\mathbf{T}^{-1}$.
5. Choose a rotation matrix $\mathbf{R}$. Predict what would happen if you operate on the original rectangle with $\mathbf{TR}$. How about $\mathbf{RT}$? Implement both of these in MATLAB and check. How would you undo each of these operations? (You will first have to adjust your definition of $\mathbf{R}$ so that it is the correct size.)
6. Predict what would happen if you operate on the original rectangle with $\mathbf{STR}$. How about $\mathbf{TRS}$? How would you *undo* each of these operations? (You will first have to adjust your definition of $\mathbf{S}$ so that it is the correct size.)
7. How would you generalize translation to 3D?

Putting it all together: Dancing Animals

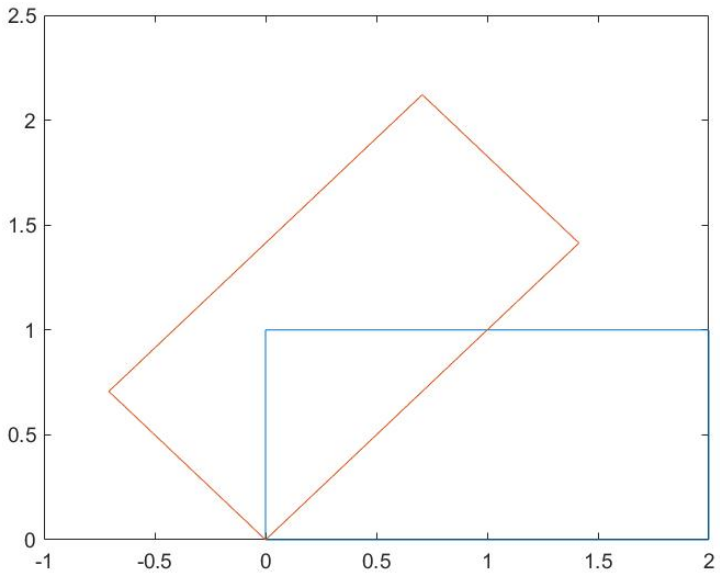
In this activity you will animate a circus act. (No real or imaginary animals will be injured in this performance.) Here is what we would like you to do:

Exercise 4.9

1. Decide on an animal.
2. Decide on a circus act that consists of a set of translations, rotations (think back to Day 2), shearings, and/or scalings in some order. Storyboard this idea and imagine the resulting animation.
3. Propose a set of points that defines the outline and relevant features of your animal. You may find `ginput` useful. Define the points in MATLAB and plot your animal.
4. Create a script that makes your animal dance (in 2-D, unless you really want to go 3-D). You may want to make use of the `pause` and `drawnow` commands.
5. Now use your sequence of operations and animate your animal! In class you will have the opportunity to show off your dancing animal!

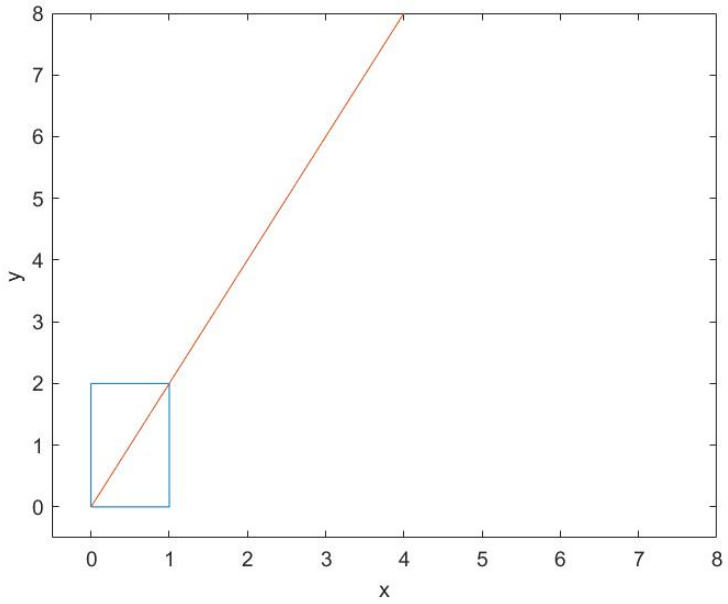
4.4 Conceptual Quiz

1. The orange shape is the result of applying a matrix $\mathbf{M}$ to the blue rectangle.



   What is the determinant of $\mathbf{M}$?

2. The orange shape is the result of applying a matrix $\mathbf{M}$ to the blue rectangle.



What is the determinant of $\mathbf{M}$?

3. The determinant is multiplicative, i.e, $\det(\mathbf{AB}) = \det(\mathbf{A})\det(\mathbf{B})$. Let $\mathbf{M}$ be a matrix such that $\det(\mathbf{M}) = \frac{1}{3}$. What's $\det(\mathbf{M}^{-1})$? (Hint: $\det(\mathbf{I}) = 1$.)

4. Let R be a rectangle with area 1. Apply the scaling matrix $\mathbf{S} = \begin{bmatrix} s_1 & 0 \\ 0 & s_2 \end{bmatrix}$. What is the area of $\mathbf{S}R$?

   A. $\frac{s_1 s_2}{2}$
   B. 1
   C. $s_1 s_2$
   D. $s_1 + s_2$

5. True or false: Any shearing matrix $\mathbf{S}$ and any rotation matrix $\mathbf{R}$ commute, i.e., $\mathbf{RS} = \mathbf{SR}$.

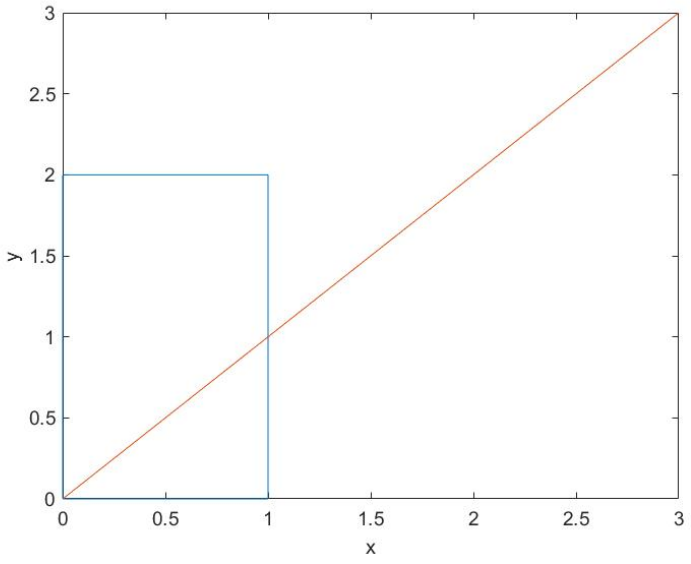
Solution 4.1

1. The determinant is 1. (Recall that $\cos^2\theta + \sin^2\theta = 1$.)
2. The determinant is -1.
3. The determinant is 1.

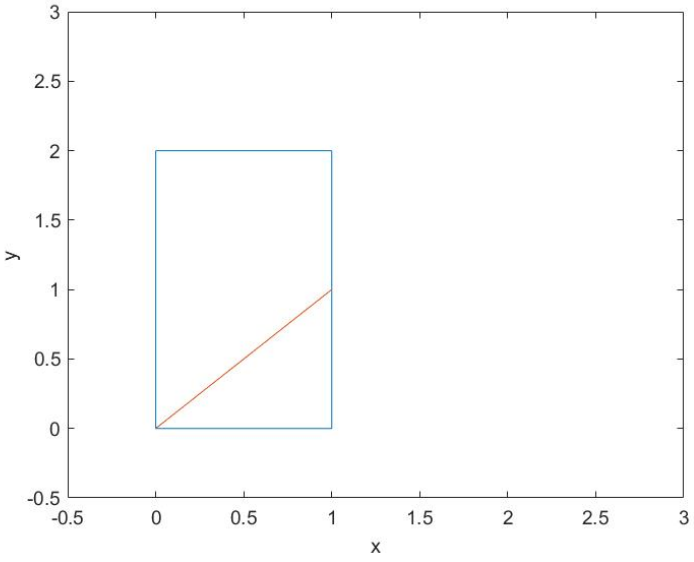
Solution 4.2

1. Each of the figures below shows the basic blue rectangle and the orange rectangle, which is the result of applying the transformation.

a)



b)



2. It is not possible to undo these matrix transformations. Since everything is squished onto the same line, we would not be able to distinguish the original vectors.

Notice that, in the above matrices, the first row is a constant multiple of the second row. In other words, the matrix looks like $\begin{bmatrix} a & b \\ ca & cb \end{bmatrix}$ for some constant c. If we apply a matrix of this form to a point in 2D space reprsented by the vector $\begin{bmatrix} x \\ y \end{bmatrix}$, then the result will be $\begin{bmatrix} z \\ cz \end{bmatrix}$, where $z = ax + by$. In other words, the resulting point will always fall on the line $y = cx$.

Solution 4.4

1. $$\mathbf{w} = \mathbf{Pu} = \begin{bmatrix} 7 \\ 17 \end{bmatrix}$$

2. $$\mathbf{Qw} = \mathbf{QPu} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

3. $$\mathbf{QP} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

   which is the identity matrix

4. The determinate of $\mathbf{P}$ is 2.

5. The determinate of $\mathbf{Q}$ is $\frac{1}{2}$.

Solution 4.5

1. a)
$$(\mathbf{PB})^{-1} = \begin{bmatrix} -17/2 & 7/2 \\ 5 & -2 \end{bmatrix}$$

   b)
$$(\mathbf{P}^T)^{-1} = \begin{bmatrix} 3/2 & -2 \\ -1/2 & 1 \end{bmatrix}$$

2.
$$\left(\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}\right)^{-1} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

$$\left(\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}\right)^{-1} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\left(\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}\right)^{-1} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}$$

Solution 4.6

1. a) It's not possible to find the numbers of oranges, apples, and pears. We have the equation
$$\begin{bmatrix} 2 & 1 & 3 \\ 4 & 3 & 3 \end{bmatrix} \begin{bmatrix} n_0 \\ n_a \\ n_p \end{bmatrix} = \begin{bmatrix} 21 \\ 45 \end{bmatrix},$$
   but we cannot take the inverse of a 2×3 (non-square) matrix.

   b) Now we have the equation
$$\begin{bmatrix} 2 & 1 & 3 \\ 4 & 3 & 3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} n_0 \\ n_a \\ n_p \end{bmatrix} = \begin{bmatrix} 21 \\ 45 \\ 14 \end{bmatrix}.$$
   So by taking the inverse of the 3×3 matrix we find that $n_0 = 3, n_a = 9$ and $n_p = 2$.

   c) The determinant of the matrix is 2.

2. a) The equation becomes
$$\begin{bmatrix} 2 & \frac{3}{2} & \frac{3}{2} \\ 4 & 3 & 3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} n_0 \\ n_a \\ n_p \end{bmatrix} = \begin{bmatrix} 21 \\ 45 \\ 14 \end{bmatrix}.$$
   But the matrix is not invertible, so we cannot solve for the number of fruit.

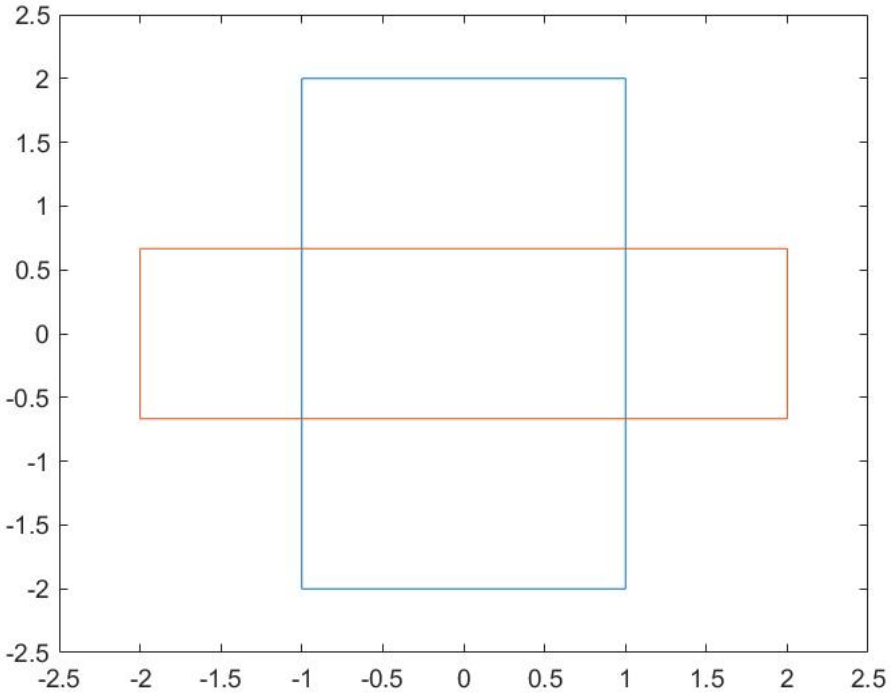
   b) The determinant of the matrix is 0.

   c)

Solution 4.7

1. The length of the rectangle would double in the x direction and be reduced to 1/3 the length in the y direction.

2. First we define the corners of the rectangle as the columns in a matrix
   » `points=[1 1 -1 -1; 2 -2 -2 2]`
   and we define the scaling matrix
   » `S=[2 0; 0 1/3]`. Then we simply multiply them
   » `scaledpoint=S*points`.
   Plotting them, here is the original rectangle in blue and the scaled rectangle in orange



3. The area is reduced from 8 units^2 to 5.33 units^2, or 2/3 of the original area.

4. To undo the process we use the inverse of the $\mathbf{S}$ matrix, or $\mathbf{S}^{-1}$ would be used.

$$\mathbf{S}^{-1} = \begin{bmatrix} 0.5 & 0 \\ 0 & 3 \end{bmatrix}.$$

   You should check that $\mathbf{S}^{-1}\mathbf{S} = \mathbf{S}\mathbf{S}^{-1} = \mathbf{I}$.

5. We define the inverse matrix » `Sinv=[0.5 0; 0 3]` and check that » `S*Sinv` and `Sinv*S` both produce the identity matrix.

6. The area of the rectangle doubles.

7. When the original rectangle is operated on with

   $$\mathbf{SR}$$

   , the resulting image will be a horizontally stretched parallelogram. When the original rectangle is operated on with $\mathbf{RS}$, the resulting image will be the scaled rectangle from the previous exercise only rotated 60 degrees counter-clockwise.

8. $(\mathbf{SR})^{-1} = \mathbf{R}^{-1}\mathbf{S}^{-1}$ or $(\mathbf{RS})^{-1} = \mathbf{S}^{-1}\mathbf{R}^{-1}$

Solution 4.8

1. $$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix}$$

2. The rectangle would be moved 2 to the right and 3 up.

3. The area of the rectangle does not change.

4. $$\mathbf{T}^{-1} = \begin{bmatrix} 1 & 0 & -2 \\ 0 & 1 & -3 \\ 0 & 0 & 1 \end{bmatrix}$$

5. If the original rectangle is operated on by $\mathbf{TR}$, the rectangle would first be rotated with respect to the origin and than translated. If the original rectangle is operated on by $\mathbf{TR}$, the rectangle would first be translated and then rotated. As rotation happens with respect to the origin, the 2 operations will not result in the same rectangle.

   To undo the operation $\mathbf{TR}$, the resulting figure should be operated on by $\mathbf{R}^{-1}\mathbf{T}^{-1}$. To undo the operation $\mathbf{RT}$, the resulting figure should be operated on by $\mathbf{T}^{-1}\mathbf{R}^{-1}$.

6. If the original rectangle is operated on with $\mathbf{STR}$, the resulting image will be of the rectangle rotated 60 degrees around the origin, translated 2 to the right and 3 up and then scaled by $\mathbf{S}$. If the original rectangle is operated on with $\mathbf{TRS}$, the resulting image will be the scaled rectangle rotated 60 degrees around the origin and then translated 2 to the right and 3 up.

   To undo $\mathbf{STR}$, the resulting figure should be operated on by $\mathbf{R}^{-1}\mathbf{T}^{-1}\mathbf{S}^{-1}$. To undo $\mathbf{TRS}$, the resulting figure should be operated on by $\mathbf{S}^{-1}\mathbf{R}^{-1}\mathbf{T}^{-1}$.

7. $$\begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Chapter 5

Day 3: Linear Independence, Span, Basis, and Decomposition

5.1 Schedule

- 0900-0930: Debrief and Dancing Animal Demos
- 0930-1000: Synthesis
- 1000-1030: Mini-Lecture: Linear Independence, Span, Basis, Decomposition
- 1030-1045: Coffee
- 1045-1210: Technical Details: Linear Independence, Span, Basis, Decomposition
- 1210-1220: Preview

5.2 Debrief and Dancing Animal Demos

- Please discuss your overnight work with your table-mates, create a set of key concepts, and a set of ideas that you are still confused by.
- Be prepared to demo your dancing animal!

5.3 Synthesis

Exercise 5.1

You should do all of these.

1. Assume the matrix $\mathbf{D}$ represents a geometrical object. What is the correct matrix expression if we want to rotate it first ($\mathbf{R}$), then scale it ($\mathbf{S}$), and finally translate ($\mathbf{T}$) it?

   A. **DRST**
   B. **TSRD**
   C. **RSTD**
   D. **DTSR**

2. What would be the correct expression in order to undo the transformation in the previous problem?

3. $\mathbf{A}$ and $\mathbf{B}$ are square, invertible matrices of the same size. Which of the following are **always** true (no matter the entries in $\mathbf{A}$ and B?

A. $(\mathbf{AB})^T = \mathbf{B}^T\mathbf{A}^T$
B. $(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}$
C. $(\mathbf{A}^T)^{-1} = (\mathbf{A}^{-1})^T$
D. $\det(\mathbf{AB}) = \det(\mathbf{A})\det(\mathbf{B})$
E. $\mathbf{A} + \mathbf{B} = \mathbf{B} + \mathbf{A}$
F. $\mathbf{AB} = \mathbf{BA}$
G. $\det(\mathbf{AB}) = \det(\mathbf{A}) + \det(\mathbf{B})$
H. $(\mathbf{AB})^T = \mathbf{A}^T\mathbf{B}^T$
I. $(\mathbf{AB})^{-1} = \mathbf{A}^{-1}\mathbf{B}^{-1}$

5.4 Linear Independence, Span, and Decomposition

Exercise 5.2

Consider two column vectors

$$\mathbf{a}_1 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, \ \mathbf{a}_2 = \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix} \tag{5.1}$$

Both these vectors lie on the xy-plane since their z components are zero. Define a new vector $\mathbf{a}_3 = c_1\mathbf{a}_1 + c_2\mathbf{a}_2$, where c_1 and c_2 are arbitrary variables. Therefore $\mathbf{a}_3$ is a linear combination of $\mathbf{a}_1$ and $\mathbf{a}_2$.

1. Does $\mathbf{a}_3$ also lie on the xy-plane?
2. Next, define a 3×3 matrix $\mathbf{A}$ whose columns are $\mathbf{a}_1$, $\mathbf{a}_2$ and $\mathbf{a}_3$. Show that the product of $\mathbf{A}$ and any 3×1 vector always lies on the xy-plane.

Exercise 5.3

Next, we will do a similar problem, but in MATLAB. Consider the following matrix:

$$\mathbf{B} = \begin{bmatrix} 1 & 1 & 3 \\ 1 & 2 & 4 \\ 1 & 1 & 3 \end{bmatrix} \tag{5.2}$$

The third column of this matrix equals the second column plus twice the first column. Hence these three vectors lie on some plane (not the xy-plane as in the previous part).

1. Open up MATLAB and using the `quiver3` command together with `hold on`, please plot the vectors corresponding to the three columns of $\mathbf{B}$, e.g., to plot the first column, type » `quiver3(0,0,0, 1,1,1);` in MATLAB.
2. Using the "rotate 3D" function on the MATLAB figure window, rotate the figure around so that it appears as if all three arrows overlap. This should indicate that the vectors lie on a plane.
3. Using `det` compute the determinant of matrix $\mathbf{B}$. Does this make sense?

The fundamental property here is that the columns of the $\mathbf{A}$ and $\mathbf{B}$ matrices are not *linearly independent*. We shall next define the idea of linearly independent vectors more formally.

- A finite set $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$ of vectors in $\mathbf{R}^n$ is said to be *linearly dependent* if there exist scalars $c_1, c_2, \ldots, c_m$ which are not all zero, such that

$$c_1\mathbf{x}_1 + c_2\mathbf{x}_2 + \ldots + c_m\mathbf{x}_m = \mathbf{0}.$$

Note that $\mathbf{R}^n$ here refers to the set of all n-dimensional vectors that are made up of real numbers. (For example, $\mathbf{R}^1$ is the real line and $\mathbf{R}^2$ is the plane.) For any value of n, $\mathbf{R}^n$ is an example of a *vector space* - we will meet different examples of vector spaces in the future. We can also express this equation using a matrix $\mathbf{A}$, whose columns are $\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_m$.

$$\begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2 & \ldots & \mathbf{x}_m\end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_m \end{bmatrix} = \mathbf{0}\,. \tag{5.3}$$

If a non-zero solution exists to $\mathbf{Ac} = \mathbf{0}$ then the set of vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ is linearly dependent. In the case of a square matrix ($n = m$), the vectors $\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_m$ are linearly dependent if and only if the $\det(\mathbf{A}) = 0$. Otherwise, the only way to satisfy the equation above is if $c_1 = c_2 = \cdots = c_m = 0$. Figure 5.1 illustrates two examples of three vectors that are in 3D space, but are linearly dependent, since in each case, all three vectors are on a plane.

Figure 5.1: Linearly dependent vectors in $\mathbf{R}^3$. (from Wikimedia Commons).

- The set of vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ is *linearly independent* if it is not linearly dependent. In other words, the set of vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ is linearly independent if

$$c_1\mathbf{x}_1 + c_2\mathbf{x}_2 + \ldots + c_m\mathbf{x}_m = \mathbf{0} \tag{5.4}$$

  only when $c_1 = c_2 = \cdots = c_m = 0$. In other words, if the only solution to $\mathbf{Ac} = \mathbf{0}$ is $\mathbf{c} = \mathbf{0}$, the set of vectors made up of the columns of $\mathbf{A}$ is linearly independent. For a square matrix this means the set is linearly independent if and only if $\det(\mathbf{A}) \neq 0$.

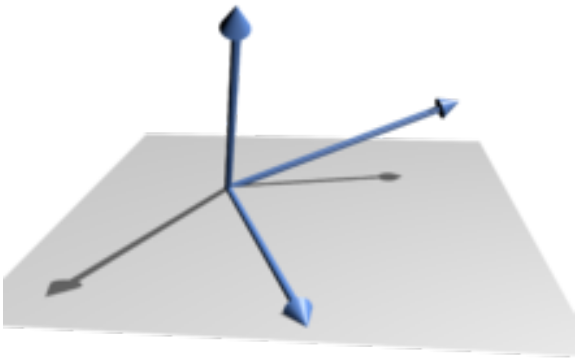


Figure 5.2: Linearly independent vectors in $\mathbf{R}^3$. (from Wikimedia Commons).

- The *span* of S is the set of all linear combinations of its vectors. In other words, the span of the set S is the set of all possible vectors of the form

$$c_1\mathbf{x}_1 + c_2\mathbf{x}_2 + \ldots + c_m\mathbf{x}_m$$

  The *span* is usually denoted by $span(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m)$.

- A finite set $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$ of vectors is said to form a basis of a vector space V, if the vectors in S are linearly independent, and every point in V can be expressed as a linear combination of the vectors in the set S. Hence, if a set of vectors S is linearly independent those vectors form a *basis* of the set which is the span of those vectors.

Let's solidify our understanding of linear dependence, bases and span by working on a few problems by hand.

Exercise 5.4

1. Determine which of the following sets of vectors are linearly independent.

   a) $\begin{bmatrix}1\\0\\0\end{bmatrix}, \begin{bmatrix}0\\1\\0\end{bmatrix}, \begin{bmatrix}0\\0\\2\end{bmatrix}$

   b) $\begin{bmatrix}1\\3\\0\end{bmatrix}, \begin{bmatrix}1\\1\\0\end{bmatrix}, \begin{bmatrix}0\\1\\0\end{bmatrix}$

   c) $\begin{bmatrix}1\\2\\3\end{bmatrix}, \begin{bmatrix}1\\1\\0\end{bmatrix}, \begin{bmatrix}3\\4\\3\end{bmatrix}$

   d) $\mathbf{p}$, $\mathbf{q}$, $\mathbf{r}$ and $\mathbf{s}$, where the vectors are all 3-dimensional.

   e) $\begin{bmatrix}1\\2\end{bmatrix}, \begin{bmatrix}3\\3\end{bmatrix}$

2. In words, describe the span of the vectors $\begin{bmatrix}1\\1\end{bmatrix}$ and $\begin{bmatrix}1\\-1\end{bmatrix}$.

3. In words, describe the span of the vectors $\begin{bmatrix}1\\1\\0\end{bmatrix}, \begin{bmatrix}2\\3\\0\end{bmatrix}$ and $\begin{bmatrix}1\\-1\\0\end{bmatrix}$ which are all in 3-dimensional Euclidean space.

Orthogonality

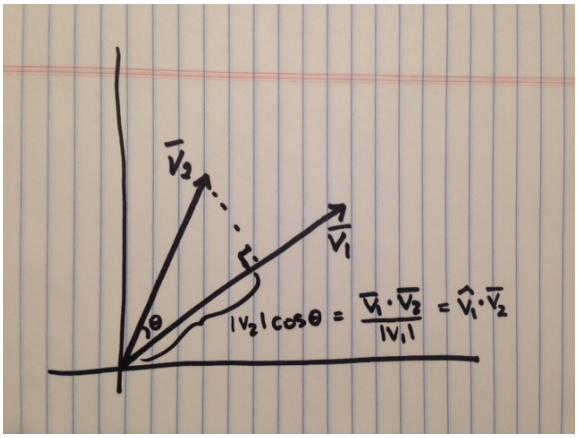


Figure 5.3: Projection

By trigonometry, if we have two vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ which have an angle of θ between them, the component of $\mathbf{v}_2$ which lies along the direction of $\mathbf{v}_1$ is $|\mathbf{v}_2| \cos\theta$. Since the dot product of the two vectors can be expressed as $|\mathbf{v}_1||\mathbf{v}_2| \cos\theta$, this component (referred to as the projection) can be written as $\mathbf{v}_1 \cdot \mathbf{v}_2/|\mathbf{v}_1|$. If the projection is zero, the vectors are *orthogonal*, and $\mathbf{v}_1 \cdot \mathbf{v}_2 = 0$. If the vectors are unit length, in addition to being normal, the vectors are said to be *orthonormal*. Additionally, if a basis set is made up of orthonormal vectors, it is known as an orthonormal basis.

A square matrix with columns of unit vectors which are orthogonal to each other is known as an orthogonal matrix. An orthogonal marix $\mathbf{A}$ has the property that $\mathbf{A}^T = \mathbf{A}^{-1}$.

Exercise 5.5

Which of the following pairs of vectors are orthogonal or orthonormal?

1. $\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \begin{bmatrix} -3 \\ 2 \\ 1 \end{bmatrix}$
2. $\begin{bmatrix} 1 \\ 0 \\ -3 \end{bmatrix}, \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix}$
3. $\begin{bmatrix} \frac{2}{\sqrt{13}} \\ \frac{-2}{\sqrt{13}} \end{bmatrix}, \begin{bmatrix} \frac{-3}{\sqrt{13}} \\ \frac{3}{\sqrt{13}} \end{bmatrix}$

Decomposition

Suppose we have a set (collection) of m basis vectors $\{\mathbf{v}_i\}$ which are normalized ($|\mathbf{v}_i| = 1$), mutually orthogonal ($\mathbf{v}_i^T\mathbf{v}_j = 0$ unless $i = j$) and span our space (every point can be written as some linear combination of the vectors $\{\mathbf{v}_i\}$). How do we actually find the linear combination which is equal to a given vector in our space?

Let's say we have a vector $\mathbf{w}$ which we are interested in expressing as a linear combination of our set of orthonormal vectors $\{\mathbf{v}_i\}$. We can write this linear combination as

$$\mathbf{w} = \sum_{i=1}^{m} c_i\mathbf{v}_i \tag{5.5}$$

and our problem is now to find the coefficients c_i in this expression.

The obvious option is to pack the basis vectors $\mathbf{v}_i$ into the columns of a matrix $\mathbf{A}$, and find solutions of

$$\mathbf{Ac} = \mathbf{w}$$

Since the columns of $\mathbf{A}$ are formed from basis vectors they are linearly independent and a non-zero solution exists and can be determined by the usual methods.

However, our basis vectors form an orthogonal set (collection) which permits a more direct calculation. Consider a particular vector $\mathbf{v}_k$ in our basis set, and let's take the dot product between $\mathbf{v}_k$ and our vector $\mathbf{w}$:

$$\mathbf{v}_k^T\mathbf{w} = \mathbf{v}_k^T\sum_{i=1}^{m} c_i\mathbf{v}_i \tag{5.6}$$

Distributing the dot product into the summation we have:

$$\mathbf{v}_k^T\mathbf{w} = \sum_{i=1}^{m} c_i\mathbf{v}_k^T\mathbf{v}_i \tag{5.7}$$

But from orthogonality we know that the dot product of any two different vectors in our orthonormal set is zero, so all terms in the sum where $k \neq i$ are zero. This leads to the following simplification

$$\mathbf{v}_k^T\mathbf{w} = c_k\mathbf{v}_k^T\mathbf{v}_k \tag{5.8}$$

In addition, since our set of vectors is normalized, we know that $\mathbf{v}_k^T\mathbf{v}_k = 1$, leaving us with

$$\mathbf{v}_k^T\mathbf{w} = c_k \tag{5.9}$$

This gives us a very nice, simple way of decomposing a vector into a linear combination of the vectors within our basis set. The dot product of each basis vector with our target vector will result in the coefficient of that term in the linear decomposition.

Exercise 5.6

1. There are many (in general, an infinite number) of bases for a given set V. Hence, we can describe elements in the set V as linear combinations of vectors from different bases. Consider

the following two basis sets which form bases for 2-dimensional space.

- $\mathbf{v}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $\mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ and
- $\mathbf{u}_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$, $\mathbf{u}_2 = \begin{bmatrix} -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$

Express the vector $\mathbf{w} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$ as a linear combination of the first basis set (i.e., a sum of scaled versions of each vector in the basis set). Repeat for the second. Please make two different drawings of $\begin{bmatrix} 2 \\ 3 \end{bmatrix}$, one expressed as a sum of scaled vectors in the first basis set and another for the vectors from the second basis set. Please label the lengths of each vector in the set.

2. Suppose that you wish to write the vector $\mathbf{w} = \begin{bmatrix} 1 \\ 2 \\ 4 \end{bmatrix}$ as a linear combination of the vectors

$$\mathbf{v}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \ \mathbf{v}_2 = \begin{bmatrix} 3 \\ 1 \\ 2 \end{bmatrix} \text{ and } \mathbf{v}_3 = \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix}.$$

Please write a matrix equation to find the coefficients of the linear combination, and solve for the coefficients using MATLAB if possible.

3. Representing vectors using different bases is a very powerful technique that we will keep coming back to in this class (in both semesters). Vectors described in different bases can give us insight that may not be so obvious when viewed in the original basis. Representing vectors in different bases can also be used for dimensionality reduction, which is an important technique that is used to speed up computations and compress data in a number of different fields. Here we will consider a problem of lossy data compression using a change of basis. Lossy compression refers to methods of representing data more efficiently, but with a loss of accuracy. Examples of lossy data compression include jpg images, and mp3 audio files. If care is taken in lossy compression, the effects of the data loss can be kept at acceptable levels (this is of course subjective and dependent on the application). We will start with a toy example and then move to more complicated ones in subsequent homework problems. Consider a set of four 2-dimensional data variables stored in the following vectors:

$$\mathbf{d}_1 = \begin{bmatrix} 2.2 \\ 1.2 \end{bmatrix}, \mathbf{d}_2 = \begin{bmatrix} 1 \\ 0.6 \end{bmatrix}, \mathbf{d}_3 = \begin{bmatrix} 1.5 \\ 0.7 \end{bmatrix}, \mathbf{d}_4 = \begin{bmatrix} 1.7 \\ 0.8 \end{bmatrix} \tag{5.10}$$

   a) In MATLAB, plot the data using points (without lines connecting them) by typing `plot([2.2 1 1.5 1.7],[1.2 0.6 0.7 0.8], 'o');` You will find that these points lie close to the line through the origin with slope 1/2.

b) Define a unit vector that points in the direction $\begin{bmatrix}2\\1\end{bmatrix}$ and call it $\mathbf{u}_1$. Find another unit vector that is orthogonal to $\mathbf{u}_1$ and call it $\mathbf{u}_2$. These vectors form a basis in 2 dimensional space.

c) Rather than storing the original data, we are now going to express the original data in terms of the new basis that we have defined. To do that, write $\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3$ and $\mathbf{d}_4$, as a linear combination of $\mathbf{u}_1$ and $\mathbf{u}_2$. You can use MATLAB here to find the coefficients.

d) In this toy example, we are going to "compress" our data by only keeping the coefficients corresponding to $\mathbf{u}_1$. i.e. we will discard the coefficient corresponding to $\mathbf{u}_2$. Suppose that we wish to recover approximations to $\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4$, from the four coefficients. These approximations, which you should denote by $\tilde{\mathbf{d}}_1, \cdots \tilde{\mathbf{d}}_4$, are all scaled versions of $\mathbf{u}_1$. In your axes from part a, please plot the points corresponding to $\tilde{\mathbf{d}}_1, \cdots \tilde{\mathbf{d}}_4$. Do you think they make good approximations?

e) We can describe how well our compressed data represents our original data. One way to do this is to calculate the difference between our original and compressed data, and call this error vector $\mathbf{f_i} = \mathbf{d}_i - \tilde{\mathbf{d}}_i$. Now, compute the size of this error using $norm(\mathbf{f_i})$ for $i = 1, 2, 3, 4$. Then, summarize the error by finding the root-mean-square (RMS) error between your approximations and the true data points. The RMS function squares the errors, takes the mean, and then takes the square root. This quantity is a single number that can be used to measure how well or poorly your compressed data represents your original data. You may find MATLAB's `norm` and `rms` functions helpful here.

This toy example illustrates that we can sometime be more efficient (albeit at the cost of some accuracy) in representing (or computing) data when it is expressed in certain bases.

Solution 5.2

1. Yes, a linear combination of two vectors which lie in the xy-plane will also lie in the xy-plane.

2. Let $\mathbf{A}$ be the matrix
$$\mathbf{A} = \begin{bmatrix} 1 & 1 & c_1 + c_2 \\ 1 & 2 & c_1 + 2c_2 \\ 0 & 0 & 0 \end{bmatrix}$$
and let $\mathbf{v}$ be an arbitrary 3×1 vector
$$\mathbf{v} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}.$$
Then the product
$$\mathbf{Av} = \begin{bmatrix} x + y + (c_1 + c_2)z \\ x + 2y + (c_1 + 2c_2)z \\ 0 \end{bmatrix}$$
lies in the xy-plane

Solution 5.3

1. Type the following into MATLAB:
```
» quiver3(0,0,0,1,1,1)
» hold on
» quiver3(0,0,0,1,2,1)
» quiver3(0,0,0,3,4,3)
```

2.

3. The determinant of $\mathbf{B}$ is zero. Recall that a matrix is not invertible if and only if the determinant is zero. This matrix is not invertible since it collapses all vectors to a plane.

Solution 5.4

1. a) They are linearly independent since they span $\mathbf{R}^3$.
   b) They are linearly dependent since the first vector is equal to the second vector plus two times the third vector.
   c) They are linearly dependent since the third vector is equal to the first vector plus two times the second vector.
   d) They are linearly dependent. You can have a maximum of n linearly independent vectors in $\mathbf{R}^n$.
   e) They are linearly independent since they do not lie on the same line.

2. The span of these two vectors is all over $\mathbf{R}^2$, i.e., a plane.

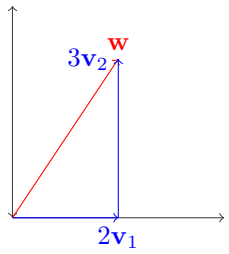
3. The span of these three vectors is the xy-plane in $\mathbf{R}^3$.

Solution 5.5

1. The dot product of these two vectors is non-zero, so they are not orthogonal.
2. The dot product of these two vectors is zero, so they are orthogonal.
3. The dot product of these two vectors is zero, so they are orthogonal. Furthermore, each vector is unit length, so they are orthonormal.

Solution 5.6

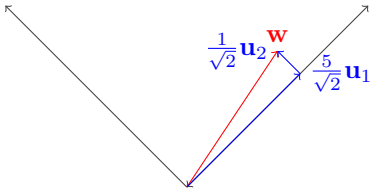
1. It's clear that $2\mathbf{v}_1 + 3\mathbf{v}_2 = \mathbf{w}$. We visualize this as



To write $\mathbf{w}$ as a linear combination of the basis vectors $\mathbf{u}_1$ and $\mathbf{u}_2$ requires a bit more work. We can set up the matrix equation

$$\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

and solve to learn that $\frac{5}{\sqrt{2}}\mathbf{u}_1 + \frac{1}{\sqrt{2}}\mathbf{u}_2 = \mathbf{w}$. We can visualize this as



2. First, we create a matrix in MATLAB whose columns are the vectors $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$,
» `V=[1 3 1; 1 1 2; 1 2 2]`
and the vector $\mathbf{w}$,
» `w=[1; 2; 4]`.
Let $\mathbf{c}$ be the vector of coefficients. We have the equation $\mathbf{V}\mathbf{c} = \mathbf{w}$, so to solve for $\mathbf{c}$ we compute $\mathbf{c} = \mathbf{V}^{-1}\mathbf{w}$. In MATLAB, we use » `inv(V)*w`. This tells us that $\mathbf{w} = -10\mathbf{v}_1 + 2\mathbf{v}_2 + 5\mathbf{v}_3$.

3. a)

   b) We define » `u1=[2; 1]` and » `u2=[-1; 2]`. There are other choices for $\mathbf{u}_2$, but they are all constant multiples of this choice, e.g., » `u2=[-2;4]`.

   c) Create a 2×2 matrix with $\mathbf{u}_1$ and $\mathbf{u}_2$ as the columns,
   » `U=[2 -1; 1 2]`
   and a 2×4 matrix the vectors $\mathbf{d}_i$ as the columns
   » `D=[2.2 1 1.5 1.7; 1.2 0.6 0.7 0.8]`.
   Then compute
   » `inv(U)*D`
   to get the matrix of coefficients. This tells us that

$$\mathbf{d}_1 = 1.12\mathbf{u}_1 + 0.04\mathbf{u}_2,\ \mathbf{d}_2 = 0.52\mathbf{u}_1 + 0.04\mathbf{u}_2,$$

$$\mathbf{d}_3 = 0.74\mathbf{u}_1 - 0.02\mathbf{u}_2, \text{ and } \mathbf{d}_1 = 0.84\mathbf{u}_1 - 0.02\mathbf{u}_2.$$

Chapter 6

Night 3: Linear Systems of Algebraic Equations

Learning Objectives

Concepts

- Determine for a system of 3 or fewer unknowns whether it has a unique solution, no solution or infinite solutions.
- Create a set of linear equations from a narrative about how the unknown variables are related to given data.
- Represent a system of linear equations with matrix, vector notation
- Solve a linear system of equations

MATLAB skills

- Compute the determinant of a matrix
- Solve systems of linear equations of the form $\mathbf{A}\mathbf{x} = \mathbf{b}$ using all three methods: inverse matrix, linsolve, or backslash operator.

Suggested Approach

See Night 1 for suggested approaches to the assignment and list of resources.

6.1 Determinants and Invertibility

You have already encountered the determinant in class: the determinant of a square matrix is a property of the matrix which among other things indicates whether a matrix is invertible or not: if the determinant of a square matrix is zero, it is non-invertible. As a reminder:

The determinant of a matrix $\mathbf{G}$ is denoted a few different ways.

$$\det(\mathbf{G}) = |\mathbf{G}| \tag{6.1}$$

For a generic 2×2 matrix $\mathbf{G}$

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

the formula for the determinant is quite straightforward:

$$\det(\mathbf{G}) = ad - bc \tag{6.2}$$

For example, for the following 2×2 matrix,

$$\det\left(\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}\right) = \begin{vmatrix}1 & 2\\ 3 & 4\end{vmatrix} = (1)(4) - (2)(3) = -2 \tag{6.3}$$

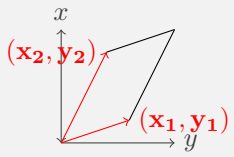
You already considered the determinant of some transformation matrices, now let's consider what the determinant is really telling us about a general matrix.

Exercise 6.1

1. Let $\mathbf{A}$ be a 2×2 matrix

$$\mathbf{A} = \begin{bmatrix}x_1 & x_2\\ y_1 & y_2\end{bmatrix}.$$

We can think of the columns of $\mathbf{A}$ as two vectors beginning at the origin and ending at the points (x_1, y_1) and (x_2, y_2), respectively. These vectors form a parallelogram, as shown here:



Show that the magnitude (i.e., absolute value) of $\det(\mathbf{A})$ is equal to the area of a parallelogram formed by the column vectors of the matrix $\mathbf{A}$.

2. What is the determinant of $\mathbf{A}$ if its column vectors are on the same line? Graphically, what happens to the parallelogram?

From this, you should get the feeling for the fact that the determinant is a measure of how co-linear the columns of $\mathbf{A}$ are: or in other words, how linearly independent the two columns are. The determinant therefore lets us know quickly if a linear system of algebraic equations has a solution, as illustrated in the following example.

Exercise 6.2

Consider the following matrix whose columns lie on the same line: the second column is simply twice the first column.

$$\mathbf{A} = \begin{bmatrix}1 & 2\\ 2 & 4\end{bmatrix} \tag{6.4}$$

1. What is $\det(\mathbf{A})$?
2. Find all the solutions to $\mathbf{Ax} = \mathbf{0}$.
3. For which vectors $\mathbf{b}$ does $\mathbf{Ax} = \mathbf{b}$ have a solution? Why are there only certain $\mathbf{b}$ vectors that lead to solutions to $\mathbf{Ax} = \mathbf{b}$?

While the formula for the determinant of a 2×2 matrix is quite straightforward, the procedures for computing the determinant of larger matrices is more difficult, but they are well known and well documented. Fortunately, MATLAB has the `det` function which computes the determinant.

6.2 Linear Systems of Algebraic Equations: Formulation and Definition

In previous classes, you've encountered a bunch of exercises where you had to operate on a vector to find another vector:

$$\mathbf{Ax} = \mathbf{b}, \tag{6.5}$$

where $\mathbf{A}$ and $\mathbf{x}$ were known, and your job was to find $\mathbf{b}$. While this is fun and, as you saw above in the rectangle exercise, can be useful, there is another related problem which is easily as important. It involves the same equation, but now you know $\mathbf{A}$ and $\mathbf{b}$ and need to find the vector $\mathbf{x}$. As we will discuss here, this problem captures the concept of a Linear System of Algebraic Equations.

One key idea in building models is the step of abstraction: going from some real-world situation to an abstracted model for the system (e.g., a set of differential equations). There are two important aspects of building such a model: first, deciding what to include or ignore, and second, deciding how to mathematically represent those things you choose to include.

One particularly common kind of mathematical framing is a set of linear algebraic equations, which can be represented by a matrix equation. A general system of m linear algebraic equations in n unknown variables $x_1, x_2, \ldots, x_n$ takes the form

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \ldots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \ldots + a_{2n}x_n &= b_2 \\ \ldots &= \ldots \\ a_{m1}x_1 + a_{m2}x_2 + a_{m3}x_3 + \ldots + a_{mn}x_n &= b_m \end{aligned}$$

where $a_{11}, a_{12}, \ldots, a_{mn}$ are known as coefficients and $b_1, b_2, b_3, \ldots, b_m$ are constants. We can write this using matrices and vectors in the form

$$\mathbf{Ax} = \mathbf{b}$$

where $\mathbf{A}$ is the $m \times n$ *coefficient matrix*, $\mathbf{x}$ is the $n \times 1$ unknown vector, and $\mathbf{b}$ is a $m \times 1$ constant vector which is known. In other words,

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}.$$

Note that "linear" here means linear in terms of the unknown variables, e.g., if x is an unknown there are only terms like ax, and no terms like $\sin(x)$, x^2, $1/x$, etc. It is often the case that you might have *coefficients* that appear to be non-linear; for example, in solving physics problems, you might have coefficients that depended on trig functions of angles, such as $(L\cos\theta)F_x$, which is is linear in F_x but not linear in θ). Be careful to be clear about what you're solving for when you decide whether something is linear or non-linear.

6.3 Using Matrix Inverses to Solve Linear Systems

Over the last week, you have worked with rotation matrices, and transformations that were compositions of simpler rotations, and learned how to invert them. When you multiply a vector by any matrix (not just ones that are associated with simple spatial transformations), you transform the original vector $\mathbf{x}$ into a new vector $\mathbf{b}$.

$$\mathbf{A}\mathbf{x} = \mathbf{b}$$

More generally (than rotations), you can *often* undo the linear transformation (just like you did with the rotation matrix). Undoing this linear transformation is a linear transformation itself! Therefore the act of undoing a linear transformation can be formulated with a matrix multiply.

$$\mathbf{A}^{-1}\mathbf{A}\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$
$$\Rightarrow \mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$

This reduces our linear system of algebraic equations problem to the problem of finding the inverse of our matrix $\mathbf{A}$. Note this is only possible if $\mathbf{A}$ is square and *invertible*.

When solving a system of equations, at least half of the battle is typically getting your system abstracted to the point that it can be thought of as a system of linear equations. The following are a set of problems. You don't need to solve these problems – you just need to formulate them as linear algebra problems.

An Investment Example

In this section we will focus on deciding whether and how you can abstract the system to a mathematical model that can be written as a matrix equation.

Exercise 6.3

Suppose that the following table describes the stock holdings of three of the QEA instructors. Also suppose that on a given day the value of the Apple, IBM and General Mill's stock are $100, $50 and

$20 respectively.

	Apple	IBM	General Mills
Jeff	100	100	100
Emily	100	200	0
John	50	50	200

1. *Here's your first linear algebra formulation question:* What is the total value of the holdings for each professor on the day in question? Can you formulate this as a matrix expression? If so, what is it? If not, why not?

2. Now, suppose that you do not know how many shares of each stock are owned by the instructors. However, you know that the total value of the stocks for each instructor for three consecutive days is as given in the following table

	Jeff	Emily	John
Day 1	$1500	$2600	$950
Day 2	$1600	$2810	$1020
Day 3	$1400	$2550	$1000

You also know that the price of each stock on each of the three days was as follows:

	Apple	IBM	General Mills
Day 1	$100	$50	$20
Day 2	$110	$50	$22
Day 3	$100	$40	$30

Now here's the second formulation question: how many stocks of each company does each professor own? Can you formulate this as a matrix equation? If so, what are the matrices/vectors? If not, why not?

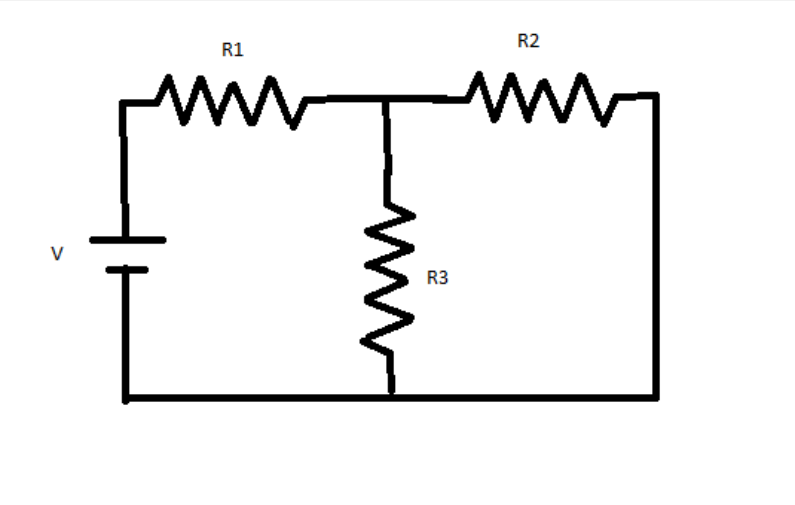
An Electrical Example

Remembering your circuit analysis back from ISIM, recall that Kirkhoff's laws:

- Kirkhoff's Voltage Law says that the sum of all the voltage drops around any loop of a circuit must sum to zero. (Batteries contribute a voltage increase of V, resistors contribute a voltage drop of IR.)
- Kirkhoff's Current Law says that the sum of all current going into and out of any junction of wires in the circuit must be zero.

Exercise 6.4

In the following circuit, consider that there is a current I_1 going through resistor R_1, a current I_2 going through resistor R_2 and a current I_3 going through resistor R_3. Find a linear algebra expression for the vector of our three unknown currents.



6.4 Types of Linear Systems and Types of Solutions

Consider the linear system of algebraic equations expressed in matrix-vector form as,

$$\mathbf{A}\mathbf{x} = \mathbf{b}.$$

If $\mathbf{b} = \mathbf{0}$ the system of linear algebraic equations is *homogeneous* and if $\mathbf{b} \neq \mathbf{0}$ the system is *non-homogeneous*. As mentioned before, we've already dealt with systems like this before when we were transforming geometrical objects, but in that case we already knew $\mathbf{x}$ and we were simply multiplying by $\mathbf{A}$ in order to get $\mathbf{b}$. Here, we are considering the so-called *inverse* problem, and trying to find $\mathbf{x}$ given $\mathbf{A}$ and $\mathbf{b}$. However, let's back up and consider some small examples to explore the solution possibilities a little.

Elimination of Variables

In high school you probably learned some basic techniques for solving small linear systems of algebraic equations. Consider the following linear system of algebraic equations,

$$2x_1 + 3x_2 = 6 \tag{6.6}$$
$$4x_1 + 9x_2 = 15 \tag{6.7}$$

The basic technique, called *elimination of Variables*, proceeds as follows: First, solve equation (2) for x_1

$$x_1 = 3 - \frac{3}{2}x_2 \tag{6.8}$$

Now substitute this expression for x_1 into equation (3)

$$4(3 - \frac{3}{2}x_2) + 9x_2 = 15$$

Now we simplify this equation

$$\begin{aligned} 12 - 6x_2 + 9x_2 &= 15 \\ \Rightarrow 3x_2 &= 3 \end{aligned}$$

and solve for x_2 to give $x_2 = 1$. Now we substitute this solution back into equation (2) or (4) to determine $x_1 = \frac{3}{2}$. The original linear system of algebraic equations therefore has a unique solution, $\mathbf{x} = \begin{bmatrix} 1 \\ 3/2 \end{bmatrix}$.

However, not all linear systems of algebraic equations have a unique solution. For example, the system

$$x_1 + 2x_2 = 1 \tag{6.9}$$
$$2x_1 + 4x_2 = 2 \tag{6.10}$$

has an infinite number of solutions because equation (6) is just a multiple of equation (5). Solving equation (5) for x_1 gives

$$x_1 = 1 - 2x_2$$

and choosing an arbitrary value of $x_2 = \alpha$ gives

$$\begin{aligned} x_1 &= 1 - 2\alpha \\ x_2 &= \alpha \end{aligned}$$

or in vector form

$$\mathbf{x} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \alpha \begin{bmatrix} -2 \\ 1 \end{bmatrix}$$

This defines an infinite number of solutions since α is any real number. What do you notice about each part of this vector?

It's also possible that a linear system of algebraic equations has no solution. For example, the system

$$x_1 + 2x_2 = 1 \tag{6.11}$$
$$2x_1 + 4x_2 = 1 \tag{6.12}$$

has no solution. Solving equation (8) for x_2 gives

$$x_2 = \frac{1}{4} - \frac{1}{2}x_1$$

and replacing into equation (7) gives

$$x_1 + 2(\frac{1}{4} - \frac{1}{2}x_1) = 1$$

which on simplification gives

$$\frac{1}{2} = 1$$

which hopefully we all agree is incorrect. We assumed that there was a solution, performed elimination and substitution and found a statement that contradicts our assumption: no solution therefore exists.

Exercise 6.5

1. Using the technique of elimination of variables described above, determine which values of h and k result in the following system of linear algebraic equations having (a) no solution, (b) a unique solution, and (c) infinitely many solutions?

$$\begin{aligned} x_1 + hx_2 &= 1 \\ 2x_1 + 3x_2 &= k \end{aligned}$$

2. Using the technique of elimination of variables described above, determine whether the following linear systems of algebraic equations have zero, one, or infinitely many solutions. If solution(s) exist, determine the actual solution(s).

   a)

$$\begin{aligned} x_1 + x_2 + x_3 &= 6 \\ x_2 + x_3 &= 2 \\ x_1 - 2x_3 &= 4 \end{aligned}$$

   b)

$$\begin{aligned} x_1 + x_2 + x_3 &= -6 \\ 2x_1 + x_2 - x_3 &= 18 \\ x_1 - 2x_3 &= 4 \end{aligned}$$

   c)

$$\begin{aligned} x_1 + x_2 + x_3 &= 6 \\ 2x_1 + x_2 - x_3 &= 10 \\ x_1 - 2x_3 &= 4 \end{aligned}$$

Solving a linear system of algebraic equations in MATLAB

Exercise 6.6

In the last class, you worked with an example of fruits in your refrigerator, and we asked you questions like how to calculate the total weight of the fruits, how many fruits there are, etc. We can use matrix operations to calculate *inverse problems* as well, as this question illustrates. Suppose that you know that you have apples and oranges in the fridge and that in the genetically engineered future, the weights of all apples are 3oz and all oranges are 4oz. Because of inflation in this genetically engineered future, the price of each apple is $1 and the price of each orange is $2. Suppose that you also know that you paid $13 total for your fruit and the total weight of the fruit is 33 oz. We can use this information and tools we have developed to figure out how many apples and oranges we have. Let n_o and n_a be the numbers of oranges and apples in your fridge respectively, and that you don't know what these numbers are. Define the following vectors

$$\mathbf{n} = \begin{bmatrix} n_o \\ n_a \end{bmatrix} \tag{6.13}$$

$$\mathbf{d} = \begin{bmatrix} 13 \\ 33 \end{bmatrix} \tag{6.14}$$

1. Write an equation relating $\mathbf{n}$ and $\mathbf{d}$, using a matrix-vector product.
2. Calculate how many oranges and apples you have.
3. Why this kind of problem is often called an inverse problem?

Exercise 6.7

1. Consider the example with the fruits that you worked out earlier. Now, in addition to apples and oranges, suppose you also had an unknown number of pears which each weigh 3 oz, and cost $3. Additionally, suppose that the total weight of the fruits is 45 oz, and you paid a total of $21 for the fruit.
   a) If possible find the numbers of oranges, apples and pears. If not, please explain why.
   b) Suppose that you additionally know that you have a total of 14 fruits. Can you formulate and solve a matrix-vector equation to find out the numbers of oranges, apples and pears you have?
   c) What is the determinant of the matrix you have set up to solve this?
2. The fruit vendors bought the pricing algorithm from Uber. Oranges are still $2, pears are now only $1.50, and (due to an influx of teachers) apples are now surging at $1.50 each. Their

weights stay the same. You return to the market, and again purchase 14 fruits, which have the same total weight and total cost.

a) Can you formulate and solve a matrix-vector equation to find out the numbers of oranges, apples and pears you have?

b) What is the determinant of the matrix you have set up to solve this?

3. Recall the example with fruits from class: Suppose that you have a total number of 14 apples, oranges and pears in your fridge. Suppose that each apple costs $1, each orange costs $ 2 and each pear costs $3. Assume also that the weights of every apple is 3 oz, every orange is 4 oz and every pear is 3 oz. Additionally, suppose that the total weight of the fruits is 45 oz, and you paid a total of $21 for the fruit.

a) Formulate (or look up your formulation from class) and write down (but don't solve it yet) a matrix-vector equation to find out the numbers of oranges, apples and pears you have.

b) Solve this equation to find the numbers of apples, oranges and pears using the following approaches (they will of course give you the same results, but we want you to get familiar with using the different operations here).

i. Using MATLAB, compute the inverse of the matrix in part a and use it to find the numbers of apples, oranges and pears.

ii. Use MATLAB's `linsolve` function to find the numbers of apples, oranges and pears.

iii. Use MATLAB's `\` operator to find the numbers of apples, oranges and pears.

6.5 Conceptual Quiz

1. Select the matrices which are invertible.

a) $\begin{bmatrix} 2 & 3 \\ 1 & 4 \end{bmatrix}$

b) $\begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}$

c) $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

d) $\begin{bmatrix} 1 & 2 \\ 4 & 8 \end{bmatrix}$

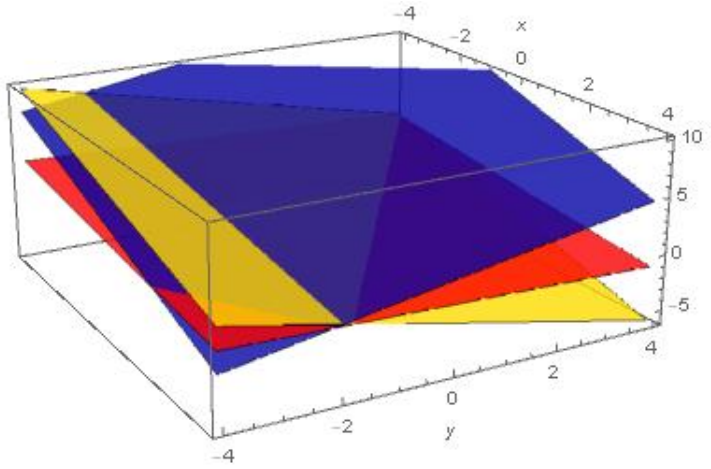
e) $\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$

2. Let (a, b, c) be the point of interesection for the following three planes, pictured below:

$$z = 2 - x - y$$

$$z = (31 - 6x + 4y)/5$$

$$z = (13 - 5x - 2y)/2$$



What is a?

3. How many solutions does the following system of equations have?

$$x + y = 9$$

$$x - z = 2$$

$$y + z = 7$$

A. Zero
B. One
C. Two
D. Infinitely many

4. What is the area of a parallelogram whose vertices are $(0, 0)$, $(2, 4)$, $(5, 1)$ and $(7, 5)$?

5. Solve the following system of linear equations

$$x - y = 2$$

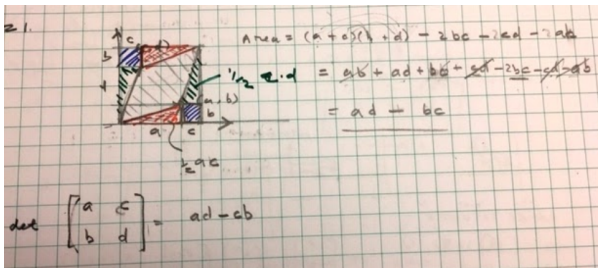
$$3x + z = 11$$

$$y - 2z = -3$$

What is the value of y?

Solution 6.1

1.

2. 

3. The determinant is equal to 0, or $\det(\mathbf{A})$=0.

Solution 6.2

1. $\det(\mathbf{A})$=(1)(4)-(2)(2)=0
2. There are infinitely many solutions of the form $-x_1 = 2x_2$.
3. Solutions are of the form $\mathbf{b} = \begin{bmatrix} k \\ 2k \end{bmatrix}$ where k is a constant.

Solution 6.3

1. This can be formulated as $\mathbf{Ax} = \mathbf{b}$ where

$$\mathbf{A} = \begin{bmatrix} 100 & 100 & 100 \\ 100 & 200 & 0 \\ 50 & 50 & 200 \end{bmatrix}, \ \mathbf{x} = \begin{bmatrix} 100 \\ 50 \\ 20 \end{bmatrix}, \text{ and } \mathbf{b} = \begin{bmatrix} d_{jd} \\ d_{et} \\ d_{jg} \end{bmatrix}.$$

   Doing the matrix multiplication shows that Jeff has $d_{jd} = 17000$, Emily has $d_{et} = 20000$, and John has $d_{jg} = 11500$.

2. There are several ways to do this. Perhaps the simplest is to compute each person's stock holding individually. To do this, we let $\mathbf{A}$ be a matrix with the stock prices

$$\mathbf{A} = \begin{bmatrix} 100 & 50 & 20 \\ 110 & 50 & 22 \\ 100 & 40 & 30 \end{bmatrix},$$

   let $\mathbf{b}_{jd}$ be a vector representing the value of Jeff's stocks on each day,

$$\mathbf{b}_{jd} = \begin{bmatrix} 1500 \\ 1600 \\ 1400 \end{bmatrix},$$

and let $\mathbf{x}_{jd}$ be a vector representing Jeff's stock holdings (i.e., the first entry tells us how many stocks of Apple he has, the second entry is IBM, and the third is General Mills). This gives the equation $\mathbf{A}\mathbf{x}_{jd} = \mathbf{b}_{jd}$. By inverting A we can solve for $\mathbf{x}_{jd}$. Then we repeat this procedure for each of the other instructors.

But... we can do it quicker! Form a 3×3 matrix $\mathbf{X}$ whose columns are made the vectors $\mathbf{x}_{jd}, \mathbf{x}_{et}$, and $\mathbf{x}_{jg}$. Then form a 3×3 matrix $\mathbf{B}$ whose columns are made of the vectors $\mathbf{b}_{jd}, \mathbf{b}_{et}$, and $\mathbf{b}_{jg}$. This gives the equation $\mathbf{AX} = \mathbf{B}$. Inverting A, we can solve for $\mathbf{X}$:

	Jeff	Emily	John
Apple	10	20	5
IBM	10	10	5
General Mills	0	5	10

Solution 6.4

$$\begin{bmatrix} R_1 & 0 & R_3 \\ 0 & -R_2 & R_3 \\ 1 & -1 & -1 \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \begin{bmatrix} v \\ 0 \\ 0 \end{bmatrix}$$

Solution 6.5

1. Rearrange the equations to linear form $y = mx + b$. If the lines are identical, there are infinitely many solutions; if the lines are parallel, but don't overlap, there are zero solutions; if the lines are not parallel, there is one solution.
   a) h=3/2, k≠2,
   b) h≠3/2,
   c) h=3/2, k=2
2. a) $x = \begin{bmatrix} 4 \\ 2 \\ 0 \end{bmatrix}$
   b) No Solution
   c) Infinite Solutions

Solution 6.6

1. $\begin{bmatrix} 2 & 1 \\ 4 & 3 \end{bmatrix} \begin{bmatrix} n_o \\ n_a \end{bmatrix} = \begin{bmatrix} 13 \\ 33 \end{bmatrix}$
2. $\begin{bmatrix} n_o \\ n_a \end{bmatrix} = \begin{bmatrix} 3 \\ 7 \end{bmatrix}$
3. In this case we know the result $\mathbf{b}$, and are working backwards to find the number of apples and oranges. We also use a matrix inverse to find the result.

Solution 6.7

1. a) No, you have three unknowns and only two equations.

   b) Yes, you now have three equations and three unknowns.

$$\begin{bmatrix} 2 & 1 & 3 \\ 4 & 3 & 3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} n_o \\ n_a \\ n_p \end{bmatrix} = \begin{bmatrix} 21 \\ 45 \\ 14 \end{bmatrix}$$

$$\begin{bmatrix} n_o \\ n_a \\ n_p \end{bmatrix} = \begin{bmatrix} 3 \\ 9 \\ 2 \end{bmatrix}$$

   c) $det(\mathbf{A}) = 2$

2. a)

$$\begin{bmatrix} 2 & 1.50 & 1.50 \\ 4 & 3 & 3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} n_o \\ n_a \\ n_p \end{bmatrix} = \begin{bmatrix} 21 \\ 45 \\ 14 \end{bmatrix}$$

   This forumation cannot be solved because A is not invertible.

   b) $det(\mathbf{A}) = 0$

3. a)

$$\begin{bmatrix} 2 & 1 & 3 \\ 4 & 3 & 3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} n_o \\ n_a \\ n_p \end{bmatrix} = \begin{bmatrix} 21 \\ 45 \\ 14 \end{bmatrix}$$

   b) i.

$$\begin{bmatrix} n_o \\ n_a \\ n_p \end{bmatrix} = \begin{bmatrix} 3 \\ 9 \\ 2 \end{bmatrix}$$

   ii.

$$\begin{bmatrix} n_o \\ n_a \\ n_p \end{bmatrix} = \begin{bmatrix} 3 \\ 9 \\ 2 \end{bmatrix}$$

   iii.

$$\begin{bmatrix} n_o \\ n_a \\ n_p \end{bmatrix} = \begin{bmatrix} 3 \\ 9 \\ 2 \end{bmatrix}$$

Chapter 7

Day 4: Linear Systems of Algebraic Equations

7.1 Schedule

- 0900-0915: Debrief
- 0915-1000: Synthesis
- 1000-1030: Applications of LSAE
- 1030-1045: Coffee
- 1045-1115: Applications of LSAE
- 1115-1200: Concept Map for Eigenfaces

7.2 Debrief

- Please discuss your overnight work with your table-mates, create a set of key concepts, and a set of ideas that you are still confused by.

7.3 Synthesis

We will increasingly use a computational tool like MATLAB to compute determinants, matrix inverses, and the solutions to linear systems of algebraic equations. In this synthesis section we will explore the theoretical foundation of these algorithms - the so-called LU decomposition.

Gaussian Elimination

The basic process of *elimination of variables* can be formalized and is known as Gaussian Elimination. Here will briefly introduce it but you can consult other sources, such as the *Gaussian Elimination* page at WolframMathWorld for more details.

Rather than writing equations, we can cast a LSAE in matrix form and perform *Gaussian Elimination* on the augmented matrix $[\mathbf{A}\ \mathbf{b}]$.

For example, the linear systems of algebraic equations

$$\begin{aligned} 2x_1 + 3x_2 &= 6 \\ 4x_1 + 9x_2 &= 15 \end{aligned}$$

can be written as the following augmented matrix

$$\begin{bmatrix} 2 & 3 & 6 \\ 4 & 9 & 15 \end{bmatrix}$$

Thinking now in terms of rows, we replace the second row with row 2 - 2 row 1 to give

$$\begin{bmatrix} 2 & 3 & 6 \\ 0 & 3 & 3 \end{bmatrix}$$

This matrix is now in so-called *echelon* form: we can find the solution to the original LSAE by first solving the equation implied by the last row and then back-substituting into the equation implied by the previous row.

Exercise 7.1

1. Set up the augmented matrix for the following example (you will recognise this from the last assignment)

$$\begin{aligned} 2x_1 + x_2 &= 13 \\ 4x_1 + 3x_2 &= 33 \end{aligned}$$

and perform *Gaussian Elimination* to reduce the augmented matrix to *echelon form*. Interpret the resulting system and determine the solution(s).

LU Decomposition

The steps used to solve a LSAE using Gaussian Elimination can also be used to *decompose* a matrix into a product of two matrices: a *lower-triangular* matrix $\mathbf{L}$ and an *upper-triangular* matrix $\mathbf{U}$. Here we will briefly introduce it but you could consult other sources, such as the *LU Decomposition* page at WolframMathWorld for more details.

In Gaussian Elimination we execute a set of row operations. In our ongoing example, we replaced row 2 with the result of row 2 - 2 row 1. This action can be neatly represented in terms of a matrix operation. Let's multiply the original matrix equation $\mathbf{Ax} = \mathbf{b}$ with the transformation matrix

$$\mathbf{M} = \begin{bmatrix} 1 & 0 \\ -2 & 1 \end{bmatrix}$$

to form $\mathbf{MAx} = \mathbf{Mb}$. Note that this transformation leaves row 1 of $\mathbf{A}$ unchanged, and it replaces the row 2 with row 2 - 2 row 1. The product $\mathbf{MA}$ is therefore an *upper-triangular* matrix $\mathbf{U}$

$$\mathbf{U} = \begin{bmatrix} 2 & 3 \\ 0 & 3 \end{bmatrix}$$

and the LSAE is now expressed as $\mathbf{Ux} = \mathbf{Mb}$. If we now multiply this expression by $\mathbf{M}^{-1}$ we obtain

$$\mathbf{M}^{-1}\mathbf{Ux} = \mathbf{b}$$

The inverse of $\mathbf{M}$ is straight-forward to write down because it "undoes" the row operations

$$\mathbf{M}^{-1} = \begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix}$$

Notice that this matrix is just a *lower-triangular* matrix $\mathbf{L}$. The LSAE now reads

$$\mathbf{LUx} = \mathbf{b}$$

We have therefore *decomposed* the original matrix $\mathbf{A}$ into the product of $\mathbf{L}$ and $\mathbf{U}$,

$$\mathbf{A} = \mathbf{LU}$$

How does this help, you might be asking? First of all, knowing the decomposition of $\mathbf{A}$ into $\mathbf{LU}$ allows us to solve the original LSAE $\mathbf{Ax} = \mathbf{b}$. Here is how.

Let's define a new vector $\mathbf{y} = \mathbf{Ux}$. Then the original LSAE can be expressed as

$$\mathbf{Ly} = \mathbf{b}$$

which is straight-forward to solve by *forward-substitution* because $\mathbf{L}$ is *lower-triangular*,

$$\begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 6 \\ 15 \end{bmatrix}$$

and the solution for $\mathbf{y}$ is $y_1 = 6$, $y_2 = 3$. We can now solve $\mathbf{Ux} = \mathbf{y}$ for $\mathbf{x}$ using *forward-substitution* because $\mathbf{U}$ is *upper-triangular*,

$$\begin{bmatrix} 2 & 3 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 6 \\ 3 \end{bmatrix}$$

and the solution for $\mathbf{x}$ is $x_1 = 1$, $x_2 = 3/2$.

Second of all, and more importantly, knowing the decomposition of $\mathbf{A}$ into $\mathbf{LU}$ allows us to solve any LSAE involving $\mathbf{A}$. Need to solve the LSAE with a different $\mathbf{b}$? No problem, just use the $\mathbf{LU}$ decomposition that you already computed and away you go. No need to redo all the steps of *Gaussian Elimination* just because $\mathbf{b}$ changed. Need to solve a LSAE for lots of different $\mathbf{b}$'s? No problem, just use the $\mathbf{LU}$ decomposition that you already computed and away you go. Finally, if you want to compute the inverse or determinant of a matrix this is easy too using LU decomposition as we show next.

There is an algorithm in MATLAB, *lu*, which does LU decomposition for you, but you should not necessarily expect to get the same $\mathbf{L}$ and $\mathbf{U}$, even for this example. (There are a variety of ways to define the $\mathbf{L}$ and $\mathbf{U}$ matrices, but this is beyond the scope of this section.)

Exercise 7.2

1. Consider the appropriate matrix from the last exercise and perform *LU Decomposition*. Check your answer by confirming that $\mathbf{A} = \mathbf{LU}$. (Please note that you perform LU decomposition on the original matrix $\mathbf{A}$, not the augmented matrix.)

Determinant

The basic algorithm for computing a determinant of $\mathbf{A}$ is to first perform LU decomposition, and make use of the following property:

The determinant of an upper-triangular or lower-triangular matrix is just the product of the diagonal entries.

We already met another property of determinants, namely that the determinant of a product is just the product of the determinants. Therefore, $det(\mathbf{A}) = det(\mathbf{L})det(\mathbf{U})$, each of which is just the product of the diagonal entries.

Exercise 7.3

1. Consider the appropriate matrix from the last exercise and find the determinant using the LU decomposition previously determined. Check your answer using *det* in MATLAB.

Inverse

The basic algorithm for computing the inverse of $\mathbf{A}$ is to first perform LU decomposition, and make use of the following idea. $\mathbf{B}$ is the inverse of $\mathbf{A}$ if it satisfies the following property

$$\mathbf{AB} = \mathbf{I}$$

The columns of $\mathbf{B}$ are just the solutions of a LSAE with a different $\mathbf{b}$. For example, in the two by two case we can solve

$$\mathbf{Ax} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

and then

$$\mathbf{Ax} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

and if we fill the columns of $\mathbf{B}$ with the solution to these LSAE we will have constructed the inverse. Since we already have the LU decomposition of $\mathbf{A}$ we simply solve each case using the technique already presented.

For example, the first column of $\mathbf{B}$ is determined as follows: First we solve $\mathbf{Ly} = \mathbf{b}$

$$\begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

to give $y_1 = 1$ and $y_2 = -2$. Now we solve $\mathbf{Ux} = \mathbf{y}$

$$\begin{bmatrix} 2 & 3 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

and the solution for $\mathbf{x}$ is $x_1 = 3/2$, $x_2 = -2/3$. This is the entries in the first column of the inverse. Repeating this process for $\mathbf{b} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ will give the second column of the inverse which now reads

$$\mathbf{A}^{-1} = \begin{bmatrix} 3/2 & -1/2 \\ -2/3 & 1/3 \end{bmatrix}$$

Exercise 7.4

1. Consider the appropriate matrix from the previous exercise and find the inverse using the LU

decomposition previously determined. Check your answer using *inv* in MATLAB.

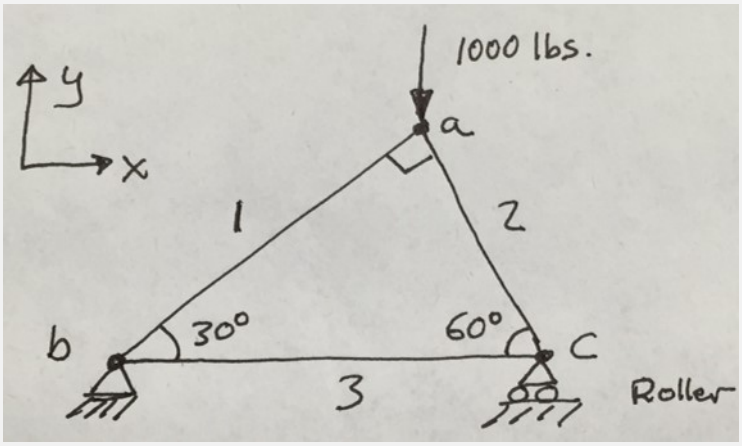
7.4 Applications of LSAE

Choose at least one of the following problems involving linear systems of algebraic equations.

Truss Analysis

Exercise 7.5

Systems of linear equations often come up in engineering when evaluating the strength and stability of structures under load. A *truss* is a simplified model of a structure. It consists of a collection of straight, rigid elements or sections that are long compared to the dimensions of their cross-section. Sections are connected only at their ends through frictionless, pin joints (Remember them? They can only constrain translation but not rotation, i.e., they can only apply force but not moments to a section). This means that sections of a truss are either in tension or compression (axial forces along its length). The roller can be assumed to be frictionless, and thus only exerts normal force. In analyzing trusses it is often assumed that the weight of the sections (dead load) is relatively small, and can therefore be neglected. The method of joints is a classic technique for determining the forces acting on all of the sections of a truss that is in static equilibrium. Here are the steps:



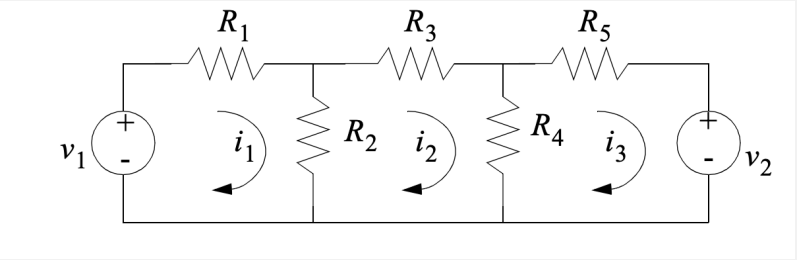
1. Draw a free body diagram for every pin in the truss. Note that the forces acting at the pin have to be in the directions implied by the things the pin is attached to!
2. Write out the equations of static equilibrium, $\sum \vec{F} = 0$, for every one of the pins. Note that some of your forces will be known forces (e.g., external loads), and some will be unknown reaction forces.

3. Express these equations in the matrix form $\mathbf{Ax} = \mathbf{b}$.
4. Evaluate whether the system is statically determinate or not. Note the connection to types of solutions to linear equations here: if you look at the form of $\mathbf{A}$, you should be able to tell whether the system is statically determinate!
5. Find the solution.

Circuit Analysis

Exercise 7.6

Systems of linear equations naturally arise in circuit analysis, although very few courses on circuits use these anymore. They do, however, form the backbone of circuit design software tools. You've met the relevant physical ideas/models before which are based on Kirchoff's circuit laws: the sum of currents into any node must be zero, and the sum of voltages around any loop must be zero. For the circuit shown in the figure:



1. Set up the linear system of algebraic equations required to solve for the three unknown currents (assuming that the resistors and the voltage sources are known.)
2. Find the solution if all of the resistors are 1 ohm, v_1 is 5 volts, and v_2 is -6 volts.

Chemical Analysis

Exercise 7.7

The complete combustion of propane, C_3H_8, with oxygen, O_2 yields carbon dioxide, CO_2, and water, H_2O. Based on conservation of mass, this reaction can be written as

$$a(C_3H_8) + b(O_2) \rightarrow c(CO_2) + d(H_2O)$$

Determine the coefficients in the combustion equation. Note that you will need to learn how to "balance" a chemical reaction.

7.5 Concept Map for Eigenfaces

For the facial recognition project we will be primarily focusing on an early facial recognition software algorithm, Eigenfaces, which is still used for face detection, and introduces some other concepts that are extremely important in both facial recognition and other tasks.

We would like you to spend some time developing an understanding of what you know, and what you don't know about facial recognition using Eigenfaces. A good way to do this is to break down the concept until you get to the point that you have terms that you *do* know:

1. Write the key term at the top or in the center. Circle it, since you don't know it.
2. Research it, and identify terms that are immediately associated with it. Write them down and connect them.
3. Circle new terms you don't understand, and break these down too.

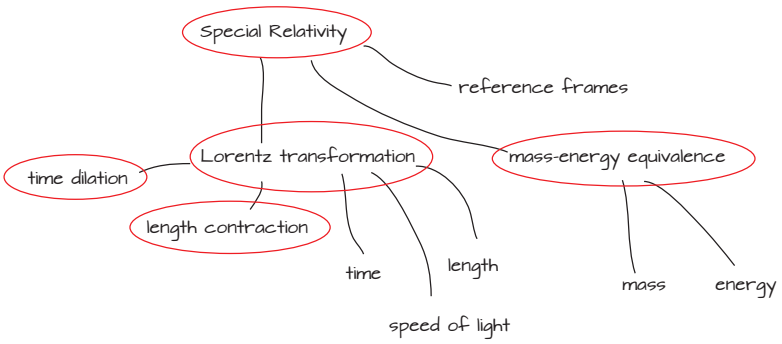


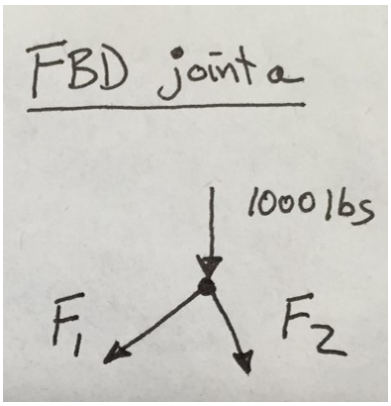
Figure 7.1: If you were trying to break down special relativity, a *portion* of your breakdown might look like this...

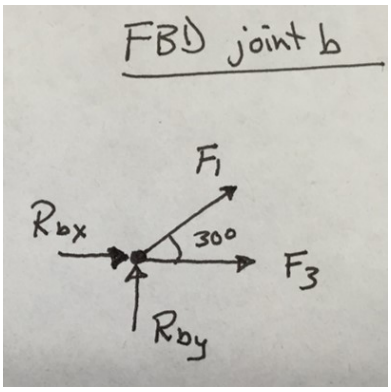
Once you've done your breakdown, try to make the following lists individually:

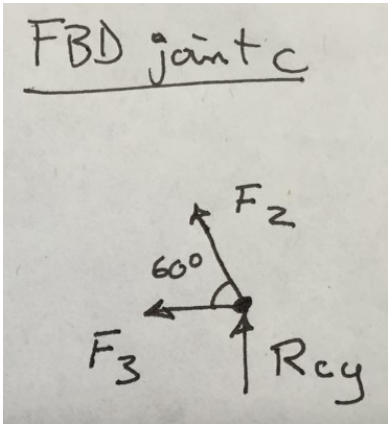
1. Relevant fundamental mathematical terms that I don't know
2. Relevant fundamental mathematical terms that I do know
3. Ideas specific to facial or image recognition that I don't know
4. Ideas specific to facial or image recognition that I do know

Solution 7.5

1. Note that the pin joint, b, cannot move in space so the ground must apply unknown reaction forces in both the x and y directions. The pin joint, c, is attached to a roller so it is free to move in the x direction but cannot move in the y direction. Therefore, the ground only applies a reaction force (unknown) in the y direction. For the entire truss, there is one known applied external force (1000 lbs) and six unknown axial and reaction forces (F_1, F_2, F_3, R_{bx}, R_{by}, and R_{cy}).







2. For each joint, $\sum F_x = 0$ and $\sum F_y = 0$. Thus we have the following six equations:

$$\begin{aligned} -F_1 \cos 30 + F_2 \cos 60 &= 0 \\ -F_1 \sin 30 - F_2 \sin 60 &= 1000 \\ R_{bx} + F_3 + F_1 \cos 30 &= 0 \\ R_{by} + F_1 \sin 30 &= 0 \\ -F_3 - F_2 \cos 60 &= 0 \\ R_{cy} + F_2 \sin 60 &= 0 \end{aligned}$$

3. These equations can be written as $\mathbf{Ax} = \mathbf{b}$ where

$$\mathbf{A} = \begin{bmatrix} -\cos 30 & \cos 60 & 0 & 0 & 0 & 0 \\ -\sin 30 & -\sin 60 & 0 & 0 & 0 & 0 \\ \cos 30 & 0 & 1 & 1 & 0 & 0 \\ \sin 30 & 0 & 0 & 0 & 1 & 0 \\ 0 & -\cos 60 & -1 & 0 & 0 & 0 \\ 0 & \sin 60 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{x} = \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ R_{bx} \\ R_{by} \\ R_{cy} \end{bmatrix}$$

,

$$\mathbf{b} = \begin{bmatrix} 0 \\ 1000 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

4. We have 6 unknowns, and 6 equations, so this is a determinate situation.

5. $$\mathbf{x} = \begin{bmatrix} -500 \\ -866.03 \\ 433.01 \\ -5.6843e^{-14} \\ 250 \\ 750 \end{bmatrix}$$

   Section 3 is the only one in tension (F_3 is positive). 1 and 2 should be in compression, and it makes sense that 2 has more compression than 1 because they support the same load, but 2 is more vertical. The horizontal reaction force is 0 because there is no net horizontal force on the system, and the sum of the vertical reaction forces is $1000\,lbf$ as we expect.

Solution 7.6

1. $$\mathbf{A} = \begin{bmatrix} -R_1 & -R_2 & 0 & 0 & 0 \\ 0 & R_2 & -R_3 & -R_4 & 0 \\ 0 & 0 & 0 & R_4 & R_5 \\ 1 & -1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 1 \end{bmatrix},$$

   $$\mathbf{x} = \begin{bmatrix} i_1 \\ i_2 \\ i_3 \\ i_4 \\ i_5 \end{bmatrix}$$

   ,

   $$\mathbf{b} = \begin{bmatrix} -V_1 \\ 0 \\ V_2 \\ 0 \\ 0 \end{bmatrix}$$

2. $$\mathbf{x} = \begin{bmatrix} 3.8750 \\ 1.1250 \\ 2.7500 \\ -1.6250 \\ -4.3750 \end{bmatrix}$$

Solution 7.7

$$C_3H_8 + 5O_2 \longrightarrow 3CO_2 + 4H_2O$$

Chapter 8

Night 4: Facial Recognition, Image Manipulation and Decomposition

Learning Objectives

Concepts

- Describe how a vector can be used to represent a data set.
- Explain how a matrix is used to represent multiple data sets.
- Explain what is meant by vectorizing a grayscale image.
- Predict the size of a vectorized image, given its pixel dimensions and color (gray or color).

MATLAB skills

- Convert a color image into a grayscale image.
- Convert an image to a matrix and back again

8.1 Ethics, Artificial Intelligence, and Facial Recognition

Face recognition is a technology with many possible applications. In just the past dozen or so years, the technology has gone from the stuff of science fiction to something that we interact with everyday (e.g., auto-tagging of images uploaded to social media). In this part of the assignment we are going to ask you to take a deep dive into how this technology manifests itself in the real world—often with mixed consequences for society.

This section is structured into three parts. First, we'll have you read about some of the issues that have been raised around face recognition technology (and more generally face analysis technology). Next, you'll read some frameworks that have been proposed to help mitigate the potential harm and maximize the benefits that might otherwise come from releasing poorly tested and biased AI systems. Finally, we'll have you branch out from face recognition technology to AI in general to examine which applications of the technology you think have the potential to most positively impact the world. You will discuss and synthesize your findings in class on Thursday, so make sure to take some sort of notes on what you read (there are also some specific prompts to respond to below).

Face Recognition Technology

Exercise 8.1

For a good overview of the issues, we'd like you to read Joy Buolamwini's written testimony that she then presented orally . You can pick whether you read the testimony or watch the video, although one nice thing about the written testimony is that it cites a lot of sources that you can read for more

information.
Based on this reading, generate a list of surprising insights (e.g., spurred by key quotes) that you gained. Also generate at least one discussion question.

Frameworks and Guidelines for Responsible Machine Learning

Exercise 8.2

Face recognition technology falls under the umbrella of machine learning. Machine learning is a field concerned with creating technologies that enable computers to learn to perform tasks automatically from experience (e.g., recognizing someone's identify from a picture of their face)—often by ingesting large training sets of labeled data. Sparked by a recognition that machine learning technologies were causing unanticipated harm in the real world, a lot of attention has been paid in recent years (both in industry and academia) to issues of fairness, accountability, and transparency. Here are two frameworks that have been created.

- Principles for Accountable Algorithms
- Google's Inclusive ML

Based on this reading, generate a list of surprising insights (e.g., spurred by key quotes) that you gained. Also generate at least one discussion question.
To get a sense of all of the conversations taking place around this topic, check out ACM's FAccT network of events.

Beyond Face Recognition

One thing that is important to mention at this point in the module is that while we are learning linear algebra and data analysis techniques within the context of face recognition, what you are learning can be applied to innumerable applications and fields of study. Even if we just stay within the realm of artificial intelligence, what you are learning now (and will learn later in the course) is the bedrock of many AI algorithms that are used in all sorts of applications. When learning about all of the issues that a technology like face recognition has, we find that students can sometimes have a tendency to move towards a nihilistic perspective on technology as a whole (e.g., all technology is bad / harmful). Critiquing technology and its role and effect in society is absolutely vital for *any* engineer. However, we contend that trying to understand how technology can be developed in a way that minimizes harm while maximizing benefit (e.g., the frameworks from the previous section) or by applying technology to problems or domains that have great potential for positive impact is also crucial. In this section, we are asking you to look into applications of image analysis (or artificial intelligence more generally) that have the potential for great positive impact on society.

Exercise 8.3

Find an article or paper about an application of artificial intelligence (it could be specifically about image or face analysis, but it need not be) that you think has the potential for great positive impact on society. Come to class ready to summarize the application and why you think it has the potential for positive impact. Unpack the notion of positive impact by specifying what the benefits (or downsides) would be of the application and who would reap them.
If you need some inspiration, here are some starting points (we are not claiming these are necessarily unambiguously positive, but they may provide some good starting points for your search).

- Automated diagnosis of cancer from medical images
- Automated, personalized education
- Optimizing energy use with artificial intelligence (more generally "Computational Sustainability")
- Sensing for driverless cars (e.g., pedestrian detection, road sign reading)
- Recognition and reading of text in a camera feed for people who are blind
- Automated wildlife monitoring via image analysis
- This one is kind of cheating. Olin 2nd year Austin Vesiliza put together a list of links to AI for social good projects that you might use for inspiration.

8.2 *Manipulating Images with Matrices*

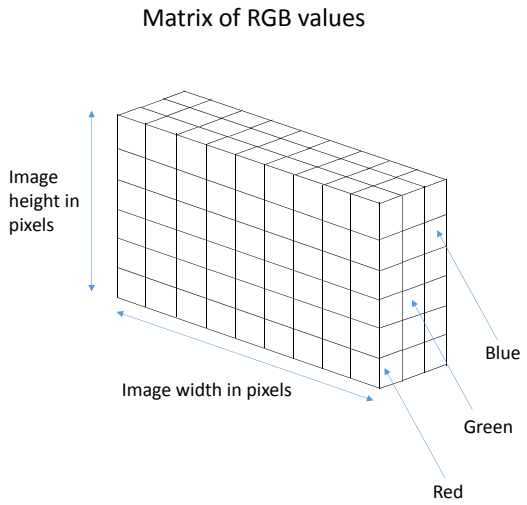


Figure 8.1: Anatomy of an RGB image array.

Exercise 8.4

Our next example is of an image pre-processing step that many of you would eventually do using built-in MATLAB functions before running your face detection algorithm.

1. Read an image file using MATLAB, and convert it to double precision numbers (the data format that MATLAB uses by default for vectors and matrices) using the following code:

   ```
   X = imread('giraffe.jpg');
   ```

   (If you get an error, try re-typing the apostrophes.)

2. Color images are stored in a 3-dimensional array (as opposed to matrices, which are 2-dimensional arrays) in MATLAB. Compare this to the smiley face image you saw in class which was a matrix whose entries are the gray-scale values. Here, instead of gray-scale values, the color information is stored in Red, Green and Blue entries of the three-dimensional array. Therefore, each pixel in the image is associated with three different values which indicate how much of Red, Green and Blue are present in that pixel. This array is illustrated in Figure 8.1.

   You can see the dimensions of this array using the following.

   ```
   size(X)
   ```

3. Display the image using

   ```
   imagesc(X);
   ```

   The image may be squashed; if you would like it not be be squashed, type `axis equal` into the command window.

4. What will the dimensions of the matrix with the grayscale representation of this image be?

5. We will now use matrix manipulations to turn the image into a grayscale image. The RGB array can be separated into three slices, one for each color. For example, the red slice is all the data in the the first layer of the array:

   ```
   X_red=X(:,:,1);
   ```

   Converting a pixel to grayscale can be accomplished by taking a linear combination of the red, green and blue values of that pixel which are weighted by 0.2989, 0.5870 and 0.1140 respectively. Use these weights to create a linear combination of the red, green, and blue slices.

6. Verify if this was done correctly by displaying the image using the following commands.

```
imagesc(grayscaleX); colormap('gray'); axis equal
```

8.3 Further Examples on Decomposition

Exercise 8.5

1. In this problem, we are going to express the temperature data for four cities we encountered earlier using a given set of basis vectors. Load some sample temperature data in MATLAB by typing » `load temperatures_and_bases.mat`. Type `whos` at the MATLAB prompt to see all your variables. You should have a matrix `T` which has the temperature data for 1 year for the cities of Boston, New York, Washington DC and Providence in that order. Use the `size` command to determine how the data are organized in this matrix. You should also have four vectors $\mathbf{u}_1 \cdots \mathbf{u}_4$ which a genie has provided to you.

   a) Verify that the vectors $\mathbf{u}_1, \cdots \mathbf{u}_4$ are all mutually orthogonal, and that they have unit length.

   b) Set up and solve the linear algebra problem in order to express each column of the temperature matrix `T` as a linear combination of $\mathbf{u}_1 \cdots \mathbf{u}_4$. Check that you can undo this operation and retrieve the original data.

   c) Now let's reconstruct an approximation to the original temperature data, using only the vectors $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3$, What is the rms error for this approximation?

   d) Compare the rms error for the previous scheme to a simpler scheme where in order to compress the data, we simply discard the temperature of Providence. When we want to reconstruct the data, we simply approximate the temperature in Providence by the temperature in Boston.

   Once again, we have to disappoint you by letting you know that there is no genie! There is just data. In the coming weeks, we are going to find out how to find bases vectors that can be useful for dimensionality reduction for a given set of random data, given some training data. This will be particularly useful in speeding up computations where instead of doing computations on all the dimensions of the data we have, we perform computations on fewer dimensions.

2. We will finally be dealing with images of faces. We are going to compress these face images in a similar way as the temperature data (we give you the bases). Here, the data have really

high dimensions (each pixel is a dimension). The bases that we give you (matrix **U**) doesn't span the entire high dimensional space (so there will be lossy compression).

a) Load the file `face_bases.mat` in MATLAB. You will see a 3-dimensional array `test_images`, of dimensions $256 \times 256 \times 424$, and a matrix U of dimensions 65536×424. The `test_images` array contains 424 grayscale images.Each image is 256×256 pixels.

b) Select any one image from the set of 424 and call it T. Display this image using » `imagesc(T); colormap('gray')`. This image is currently represented as a 256×256 matrix of grayscale values. We will find it very convenient to work with vectors instead of matrices representing an image. Therefore, to make our lives simple, we will take the data for an image which is stored in a matrix and store it in a vector.

We are going to *vectorize* this image by stacking its columns one on top of another to create a single vector that is $(256)^2 \times 1$, i.e. 65536×1 which will be a lot easier to work with. This operation can be accomplished in MATLAB as follows: » `Tstacked = reshape(T, 65536,1);`. When you need to recover the unstacked version of the image, you can undo-the stacking as follows: » `Tunstacked = reshape(Tstacked, 256, 256)`.

c) The matrix U contains a set of 424 65536×1 linearly independent vectors provided by the genie. Approximate the `Tstacked` vector as a linear combination of the first 10 of columns of U, and call this vector `Tapprox10`. `Tapprox10` should be a 65536×1 vector, and you will only have 10 weight values to find this approximation. See how well this approximation works by reshaping `Tapprox10` into a 256×256 matrix and displaying it using `imagesc` and `colormap('gray')`.

d) Now repeat the previous exercise with the first 50 columns of U and then again with the first 100 columns of U.

You should observe that the more columns of U you use, the better the approximation. Note here that we are trying to approximate a 65536 dimensional vector using 10, 50 and 100 numbers. Therefore, you should not expect the approximation to work super well, but with 100 columns of U, you should be able to recognize the picture. At a later date we will quantify the fidelity of the approximation.

Note that more sophisticated image compression algorithms use methods that rely on special properties of images and human vision in order to a achieve high degree of compression.

8.4 Data: Many Measurements of the Same Thing

One of the simplest forms of data is a set of data which represents many measurements of nominally the same thing. Depending on what the goal is of our analysis, this might encompass measurements of the

same quantity across many different situations, or many instances of the same situation.

Visualizing Measurements of the Same Thing

It's usually a good idea to *look* at data before you start calculating things associated with it.

You've surely encountered these ideas before, but for the sake of completeness, we'll highlight a couple of ideas here. If you have a large number of data points (say, for example, that you measured the heights of a bunch of different people), you might choose to simply plot the data versus the person number – the index. Note here that the data is plotted as individual points, since each point represents a measurement. Ideally we might also include error bars here to indicate our uncertainty in a given measurement, but for now, let's leave that out.

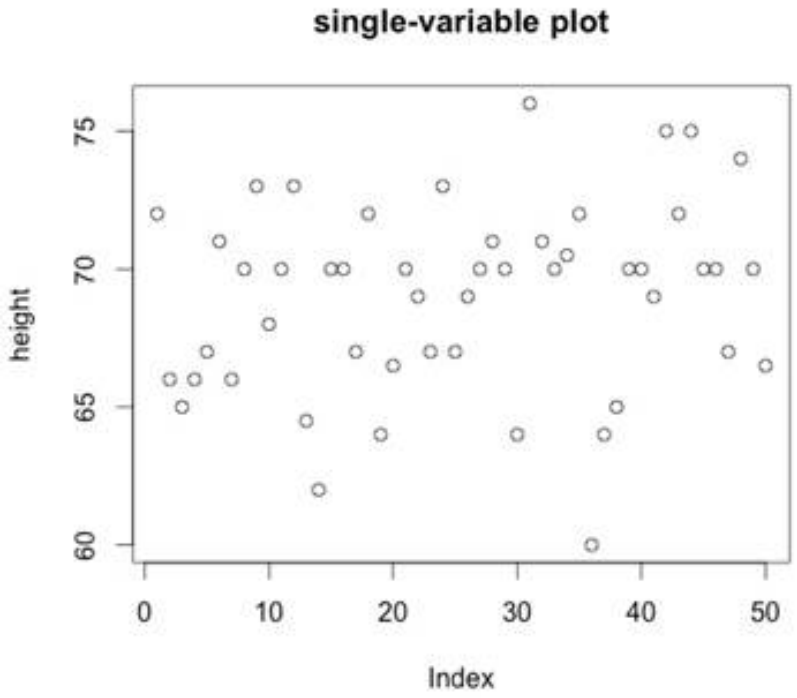


Figure 8.4: An example of a single variable plot

Alternatively, you could also visualize many measurements of the same thing by creating a *histogram*. This is a representation of how many measurements fall into different "bins": the height of a given bar is the number of samples that fall within the range associated with the bar. For example, in the figure, you can see that about 20 million people made between 0 and $5000 in 2008. You've likely seen this kind of thing before as well: it's not an uncommon way to represent test scores.

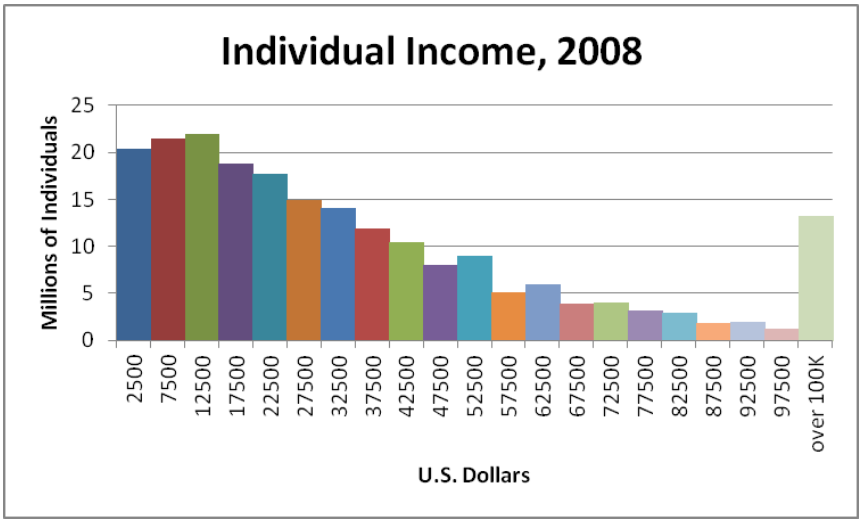


Figure 8.5: An example of a histogram.

Note, of course, that how a histogram *looks* depends on what you choose for the bins - both how many there are, and where they are centered!

Common Figures of Merit for the Same Thing

While looking at the data is certainly helpful, we can also extract or calculate a couple of important figures of merit of the data. The first is the average, or *mean* of the data, given by summing all the elements in the dataset $\{d_i\}$ and dividing by the number N of elements in the set:

$$\mu = \frac{1}{N}\sum_{i=1}^{N} d_i \tag{8.1}$$

Note that if our data is a continuous function $f(x)$ over a range of the independent variable x as opposed to a set of discrete points, we can express the same thing as an integral:

$$\mu = \frac{\int_{range} f(x)dx}{\int_{range} dx} \tag{8.2}$$

The average captures the center or 'expected value' of the distribution of data. In addition to this, it is often helpful to capture the spread of the data around this average. There are a few different metrics which are used for this. A simple one is the *variance* from the mean, σ^2: the average of the squared difference between each data point and the mean.

$$\sigma^2 = \frac{1}{N-1}\sum_{i=1}^{N}(d_i - \mu)^2 \tag{8.3}$$

Please note that this definition normalizes using $N-1$, but you will often see alternative definitions which normalize using N. Another commonly encountered measure is the *standard deviation*, which is simply the

square root of the variance from the mean:

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(d_i - \mu)^2} \tag{8.4}$$

Exercise 8.6

1. Look at the single variable plot in Figure 8.4 above. Estimate the value of the mean and the value of the standard deviation. What are the units of each?
2. Look at the histogram plot in Figure 8.5 above. Estimate the value of the mean and the value of the standard deviation.
3. What is the mean and standard deviation of this data set (Do this in your head!)

$$\{1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3\}$$

4. Begin by considering the simple dataset of the high temperatures in Needham for ten days in March:

$$T = \{57, 61, 46, 43, 46, 46, 54, 46, 46, 55\} \tag{8.5}$$

   a) By hand, create a histogram of this data. What size bin makes sense? What bin centering makes sense?

   b) By hand, compute the mean temperature over these ten days. If you look at the data, does this mean make sense?

   c) By hand, compute the variance and standard deviation of the temperature over these ten days. If you look at the data histogram does this make sense?

   d) This dataset has a flaw: it has a small number of datapoints. What do you see as the possible effects of having such a small sample?

5. Now consider the larger dataset below of the approximated heights of the Olin faculty, measured in inches. In MATLAB, create a vector which has this dataset as the entries.

$$\begin{aligned} H = \{&63, 66, 71, 65, 70, 66, 67, 65, 67, 74, 64, 75, 68, 67, 70, 73, 66, 70, 72, 62, 68, \\ &70, 62, 69, 66, 70, 70, 68, 69, 70, 71, 65, 64, 71, 64, 78, 69, 70, 65, 66, 72, 64\} \end{aligned}$$

   a) Computationally histogram this data. What size bin makes sense? What bin centering makes sense? Try a few different combinations. See MATLAB function `histogram`.

   b) Computationally, find the mean, standard deviation, and variance of this dataset. See MATLAB functions `mean`, `std`, and `var`.

   c) Does the mean, standard deviation, and variance make sense given the histogram of the data?

8.5 Brightness and Contrast

The brightness and contrast of images is controlled by scaling the histogram of the pixel values. Try this out!

Note: for displaying images in this part, make sure to NOT use imagesc: imagesc is specifically setup to auto-scale the image to use the full range from 0 to 255. Just use the command 'image'.

Exercise 8.7

1. Load an image of your choice into MATLAB using the `imread` command. (Make sure you are in the correct directory for the image or give it the complete path). Display the image using the 'image' command.
2. If your image is a color image, convert it into grayscale by using the `rgb2gray` command.
3. Create a vector of the intensities in your image: use the `reshape` command to create a giant column vector in which the first n elements are the first column of the image, the next n are the second column, etc.
4. Make a histogram of the intensity values in your image. Note that the default variable type for image data is uint8 (8-bit unsigned integer) which is an integer that ranges from 0 to 255. Does your image use the entire range of values from 0 to 255? What is the minimum pixel value used? What is the maximum?
5. Find the mean of the intensities in your image data. Find the standard deviation. Is the intensity data well-centered on the available range? The location of the intensity data in the range determines the brightness of the image. How does the standard deviation compare to the available range? Does the intensity data span a good portion of the available range? This affects the contrast.
6. To adjust the brightness of your image, you can scale all of the intensity values by a multiplicative factor down (towards darker values) or up (towards brighter values). Based on looking at the histogram, should your image be brightened? Dimmed? Why?
7. To adjust the contrast, you make a linear mapping of the existing range onto the full 0 to 255 range. In other words, if you think of the current intensity value as your independent variable x, and the new intensity value as the dependent variable y, a contrast adjustment is defined by a function $y = f(x)$. Propose an equation for a line which gives you the "best" range of y's, given the input intensity values in the image. You should be able to justify this based on the histogram of the image. Note that any values of y that end up below 0 should be interpreted as 0, and any values over 255 should be interpreted as 255.
8. Implement brightness and contrast adjustment:

a) Load a picture of a face.
b) Analyze the intensity histogram.
c) Calculate the adjusted face by applying both brightness and contrast adjustments to make it as "good" as possible.
d) Create a figure that includes four subplots: the original image, the original intensity histogram, the new image, and the new intensity histogram.

9. What would happen if the function for contrast adjustment was not linear? Why might you choose a non-linear function for this mapping?

Solution 8.4

1.

2. The size of the array is $740 \times 740 \times 3$.

3. You should see the following picture:

Figure 8.2: Giraffe

4. The gray-scale version of this image is represented by a 740×740 matrix.

5. Create the matrix **grayscaleX** which represents the grayscale version of this image using » `grayscaleX = 0.2989*X(:,:,1) + 0.5870*X(:,:,2) + 0.1140*X(:,:,3)`.

6. You should see the following image:

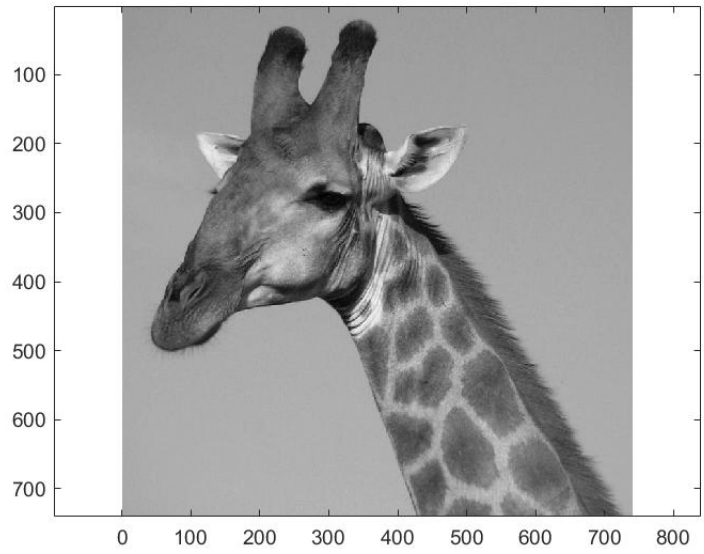


Figure 8.3: Gray Giraffe

Solution 8.5

1. a) To check that `u1` and `u2` are orthogonal, type » `transpose(u1)*u2`. The result should be zero. Check the other five pairs of vectors.
   To check that `u1` has unit length, type » `transpose(u1)*u1`. The result should be one. Check the other three vectors.
   b) First we need to construct a matrix U with columns each of the $\mathbf{u}_i$
   » `U=[u1 u2 u3 u4],`
   and convert T to the basis of $\mathbf{u}_i$ vectors by multiplying » `Tu=U*T.` You can recover the original data with » `inv(U)*Tu`.
   c)
   d)
2. a)
   b)
   c) Isolate the first 10 columns of U using » `U10=U(:,1:10);`. Then determine the weights for each of these column vectors using
   » `Tweights10=transpose(Tstacked)*U10;`
   and then take the linear combination
   » `Tapprox10=U10*transpose(Tweights10);`. Then we unstack the vector into a matrix
   » `Tapprox10unstacked=reshape(Tapprox10,256,256);` and display the image.

d)

Solution 8.6

1. Assuming that the "heights" plotted are heights of randomly selected humans, then the unit for the mean and standard deviation is inches.

2.

3. Since half the digits are 1 and the other half are 3, the mean will be the average of 1 and 3, so $\mu = 2$. Looking at the formula for standard deviation, we can see that $d_i - \mu = 1$ for each data point, so $\sigma = \sqrt{20/19}$.

4. a)
   b) We compute $\mu = 50$.
   c) We compute $\sigma = 6.15$.
   d)

5. a) By simply entering » `histogram(H)`, MATLAB automatically chooses bins of size one.
   b) Using MATLAB we find that $\mu = 68.1429$, $\sigma = 3.5721$ and $\sigma^2 = 12.7596$.
   c)

Chapter 9

Day 5: LSAE, Brain data, and C&E

9.1 Schedule

- 0900-0915: Table Dynamics Discussion
- 0915-1010: Debrief and Synthesis
- 1010-1025: Coffee Break
- 1025-1035: Correlation
- 1035-1135: Brain Data with Samantha Michalka!
- 1135-1220: AI and Society Discussion
- 1220-1230: Day Survey

9.2 Table Dynamics Survey Discussion

9.3 Debrief and Synthesis

- Please discuss your overnight work with your table-mates, and get help with the ideas that you are still confused by.

In the next set of exercises we will explore a common method for finding "the" solution of a linear system of algebraic equations ($\mathbf{Ax} = \mathbf{b}$) in the case where there are more equations that unknowns (more rows than columns). We will first need to synthesise some previous ideas about the span of vectors.

Range of $\mathbf{A}$

We discussed earlier the concept of the **span** of a collection of vectors. Recall that the span of a collection of vectors is the set of all linear combinations of the vectors. Now we will apply this concept to the columns of a matrix:

Definition: The Range of a matrix $\mathbf{A}$ *is the span of its columns.*

Exercise 9.1

Describe in words the Range of the following matrices:

1. $\mathbf{A} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

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

Exact Solution to $\mathbf{Ax} = \mathbf{b}$

When does a linear system of algebraic equations, $\mathbf{Ax} = \mathbf{b}$, have a solution? Since the product $\mathbf{Ax}$ is a linear combination of the columns of $\mathbf{A}$, then $\mathbf{Ax} = \mathbf{b}$ will have a solution if and only if $\mathbf{b}$ is in the Range of $\mathbf{A}$. Think about that, and complete the following exercise.

Exercise 9.2

Which of the following linear systems of algebraic equations will have a solution? Think about it from an equation perspective and the Range of $\mathbf{A}$ perspective.

1. $\mathbf{A} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$
2. $\mathbf{A} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$
3. $\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix} 3 \\ 7 \\ 11 \end{bmatrix}$
4. $\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix} 3 \\ 7 \\ 5 \end{bmatrix}$

Approximate solution to $\mathbf{Ax} = \mathbf{b}$

You should have found that some of these systems do not have a solution in the usual sense, i.e. there is no vector $\mathbf{x}$ which makes the equation $\mathbf{Ax} = \mathbf{b}$ true. We might refer to such a solution as an **exact** solution. We will now consider an **approximate** solution, i.e. a vector $\mathbf{x}$ which approximately satisfies

$\mathbf{Ax} = \mathbf{b}$. We will consider a particular approximation based on **orthogonal projection** now, and later in this module we will look at this approximation from a different perspective where it is known as the **Least-Squares** approximation. We met orthogonal projection earlier in the module when we spoke about vector components and basis vectors.

Exercise 9.3

Hold your hand up in front of you, and think about it as occupying a location in 3D.

1. Point to the location on the front wall that is closest to your hand.
2. Point to the location on the floor that is closest to your hand.
3. Have one of your table-mates hold a mobile white board at some angle. Now point to the location on the white board that is closest to your hand (you might have to imagine a larger mobile white board).
4. What is the relationship between the "pointing" vector and the surface being pointed at?

Now let's put this in the context of solving $\mathbf{Ax} = \mathbf{b}$.

- If $\mathbf{b}$ is not in the Range of $\mathbf{A}$ then we will define an approximate solution by orthogonal projection of $\mathbf{b}$ onto the Range of $\mathbf{A}$.
- The "pointing" vector from $\mathbf{b}$ to the relevant point in the Range of $\mathbf{A}$ is $\mathbf{Ax} - \mathbf{b}$.
- Since the Range of $\mathbf{A}$ is defined by the span of the columns of $\mathbf{A}$ then the "pointing" vector must be orthogonal to **every** column of $\mathbf{A}$.
- This implies that $\mathbf{A}^T(\mathbf{Ax} - \mathbf{b}) = \mathbf{0}$. (Think about why this must be true).
- Re-arranging this equation leads to $\mathbf{A}^T\mathbf{Ax} = \mathbf{A}^T\mathbf{b}$. The matrix $\mathbf{A}^T\mathbf{A}$ is a square matrix (which we will meet again and again this module).
- This is a linear system with equal numbers of equations and unknowns and can therefore be solved using our usual techniques. Did you get that? You should re-read this paragraph a few times. To summarize:

The approximate solution to $\mathbf{Ax} = \mathbf{b}$ *based on orthogonal projection can be obtained by solving*

$$\mathbf{A}^T\mathbf{Ax} = \mathbf{A}^T\mathbf{b}$$

This solution is also known as the ***least-squares*** *solution because it minimises the distance between* $\mathbf{b}$ *and the Range of* $\mathbf{A}$ *(more about this later in the module).*

Warning: *Do not think about* $\mathbf{x}$ *defining a coordinate system that* $\mathbf{b}$ *lives in! When you draw a picture you should think about the space that the columns of* $\mathbf{A}$ *live in. We are projecting* $\mathbf{b}$ *onto a basis defined by the columns of* $\mathbf{A}$*. The solution vector* $\mathbf{x}$ *is better thought of as a set of "weights" or "coordinates" with respect to this basis.*

Exercise 9.4

1. Consider the linear system $\mathbf{Ax} = \mathbf{b}$ where $\mathbf{A} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$. (You've already thought about this earlier).
   a) Sketch the Range of $\mathbf{A}$ and locate the point in the Range that is closest to $\mathbf{b}$.
   b) Multiply both sides of $\mathbf{Ax} = \mathbf{b}$ by $\mathbf{A}^T$ and solve the resulting linear system.
2. Consider the linear system $\mathbf{Ax} = \mathbf{b}$ where $\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix} 3 \\ 7 \\ 5 \end{bmatrix}$. (You've already thought about this earlier).
   a) Sketch the Range of $\mathbf{A}$ and locate the point in the Range that is closest to $\mathbf{b}$.
   b) Multiply both sides of $\mathbf{Ax} = \mathbf{b}$ by $\mathbf{A}^T$ and solve the resulting linear system.

Solving $\mathbf{Ax} = \mathbf{b}$ *in Matlab*

In many ways Matlab makes life easy for us. There is a single command in order to solve a linear system $\mathbf{Ax} = \mathbf{b}$

```
>> x = A\b
```

although it can also be used by typing

```
>> x = mldivide(A,b)
```

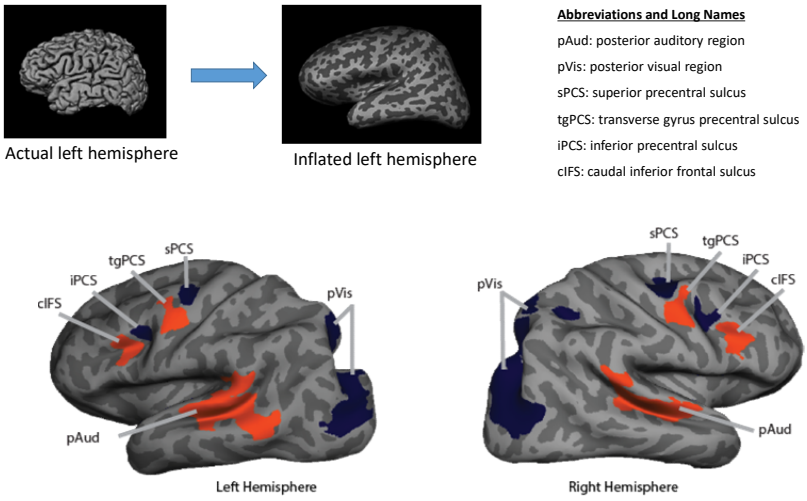
If there are more rows than columns then Matlab finds the approximate solution we discussed above. If there are equal numbers of rows and columns then Matlab computes a solution by LU decomposition. If there are less rows than columns then Matlab computes one of the infinite number of solutions - the solution it computes is not an approximation but it does select the solution that minimizes the length of the solution vector.

9.4 *Correlation (activity sheet will be printed available at your table)*

9.5 *Analyzing brain data with special guest star Sam Michalka!*

In this section, you are going to find the correlation between 12 brain regions and use this to support the hypothesis that these regions are part of two different "networks." Brain regions are considered to be part of a network if the signals in the regions are correlated with each other. Here, we are going to look at the correlations of really slow changes over time (several seconds).

We have already collected the data using functional magnetic resonance imaging (fMRI), which looks at changes in the amount of oxygen in the blood in the brain. These changes in oxygen levels are related to large populations of neurons firing and changes slowly and with several seconds of delay. Here, we are going to refer to the signal that the fMRI detects as "activity" in the brain (this is a gross oversimplification of what's going on, but is enough to approach this analysis with some guidance. Feed Sam coffee if you want to know more.)

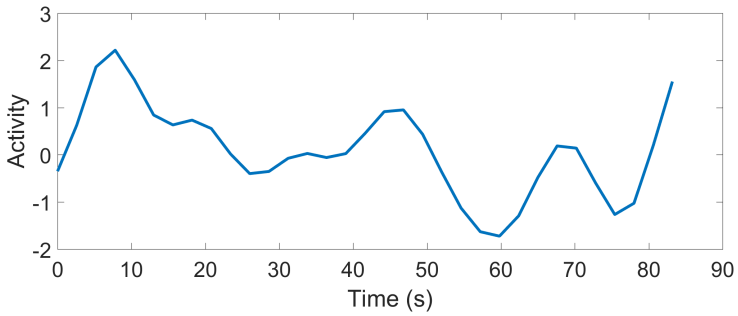


The fundamental idea behind what we are looking for is as follows:

- The brain can be divided into different regions (physical areas of the brain that are thought to serve some shared purpose). The brain is approximately symmetrical and has a left and a right hemisphere.
- We have identified 12 brain regions to investigate: 6 in the left hemisphere and 6 in the right hemisphere. They all have long fancy names (such as left superior precentral sulcus) and abbreviations (Left_sPCS). Due to the symmetry of the brain, there are matching names in each hemisphere (Right_sPCS and Left_sPCS both exist).
- The 12 brain regions that were selected were chosen because a previous experiment indicated that these regions are part of two different networks: one network that is more involved in processing auditory (sound) information and one network that is more involved in processing visual information.

- Our goal is to look at the correlation between the activity in each of these regions and the other 11 regions. Based on our first study, we hypothesize that the regions in blue will be correlated with other regions in blue to form a "visual network" and the regions in orange will be correlated with other regions in orange to form an "auditory network".

We have given you the data from one person (the actual study had more people). Also, we are not going to worry about statistical testing here; we are just going to look at the correlations in relation to each other.



Let's analyze some brain data! The plot below shows the first 80-ish seconds of data from the "right posterior auditory region" or "Right_pAud".

Exercise 9.5

1. **Consider your data and hypothesis.** Before you transform into a Matlab-mastermind, let's take a minute to think and work on the whiteboard.
   a) Roughly sketch a signal like the one shown in the plot above and label it "A". Don't worry about the details of this signal, just draw some bumps at approximately the same places.
   b) Sketch a line for a signal that has a high (but not perfect) positive correlation with "A". Label it "B". (A different color for each signal would be nice, if convenient.)
   c) Sketch a line for a signal that has a high (but not perfect) negative correlation with "A". Label it "C".
   d) Sketch a line for a signal that is uncorrelated with "A". Label it "D".
   e) Make a rough estimate of the correlation between each of these pairs of signals. Organize these estimates in a 4 x 4 table and add labels.
   f) Discuss alternate signals that you could have drawn for B, C, and/or D.
2. **Load and explore the data.** Whenever you encounter a new data set, it's helpful to look at the variables and make a few quick plots.
   a) Load the fmridata.mat file.
   b) Make sure you have the proper variables.

- There should be 12 brain region vectors, each starting with the word "Left" or "Right" (e.g., `Left_sPCS`). Each value in the vector contains a measurement at a particular point in time. The data were collected for a period of 11 minutes and 5.6 seconds. Measurements were taken every 2.6 seconds. A timepoint refers to one of those 2.6 second increments. How many timepoints do you expect? Does this match your vector length?
- The variable `braindata` contains all 12 brain region vectors organized into an array with dimensions `timepoints` × `brain regions`.
- The variable `names` contains the names of the 12 brain regions in an order that matches `braindata`.
- The variable `uncleandata` is similar to `braindata` and can be ignored for now.

c) Generate a plot with time on the x-axis and "activity" (the data values) on the y-axis. Represent each brain region as a line. You will first need to generate a new vector called `time` that represents the appropriate time increments (see above). Do your data seem reasonable? How do you know?

3. **Focus on a few pairs of regions.** The plot of all brain regions over all time points had a lot of information, so let's zoom in a bit and tie things back to the question at hand. Are the "auditory" (orange) regions correlated with other "auditory" regions? And are they correlated with the "visual" (blue) regions? Let's use one "auditory" regions called `Left_pAud` to investigate. The region `Left_pAud` is in the left hemisphere of the brain and should be colored orange in the image of the brains above.

   a) Let's begin looking the relationship between `Left_pAud` and `Right_pAud`.
   b) Look at the brain image to determine if `Right_pAud` is an "auditory" (orange) or "visual" (navy blue) type of region. Do you expect the activity of this region to be correlated with `Left_pAud`?
   c) Plot the signals from these two regions (with time on the x-axis). You may want to zoom in to the first 100 seconds to get another view. Guess the correlation between these regions.
   d) Calculate the correlation between between the two regions using the built-in MATLAB function `corr` (remember, you can get more information on a MATLAB command by typing, for example `doc corr` or `help corr` into the MATLAB command prompt).
   e) Is this correlation what you expected? How does it fit into our investigation of the correlation between "auditory" and "visual" regions?
   f) Repeat the steps above to look at the relationship between `Left_pAud` and `Left_sPCS`.
   g) Repeat the steps above to look at the relationship between `Left_pAud` and `Left_tgPCS`.

4. **Bonus fun: Generalize your analysis to calculate and display the correlations between all pairs of brain regions.** We looked at a few specific pairs of brain regions, but now we want to investigate our hypothesis by looking at all possible pairs of brain regions.

a) Create a matrix that contains the values of the correlation between each brain region. This is the same idea as the table of correlations that you wrote on the board. (Hint: Use the `braindata` matrix instead of the individual brain region vectors.) You can confirm that your values are correct by comparing to the correlations that you calculated above.

b) Display this matrix using `imagesc`. The following code may be helpful in labeling the plot:

```
>> colorbar; colormap('jet'); caxis([-1 1]);
% Set color bar parameters
>> xticks(1:12); xtickangle(-45); xticklabels(names);
% Label x axis
>> yticks(1:12); yticklabels(names);
% Label y axis
```

c) Discuss your observations from this plot. Refer to the figure with the brain image to determine which brain regions are "auditory" regions and which are "visual" regions? What patterns to you observe about their correlations?

5. **More Bonus fun: Explore the effects of preprocessing and outliers.** If you have extra time, check out the data in *uncleandata*. This data has the same structure as *braindata*, but contains some "bad" time points (208, 209, 210 , 232) caused by a blip in the recording equipment (this is a real problem that people face, though this example is extreme).

a) Plot this data versus time. How does this compare to the "clean" data from `braindata`?

b) Using the `uncleandata`, recreate the correlation matrix figure that you created using imagesc(), which shows the correlations between each pair of regions. How does this compare to your original version of this figure? What is important about these differences?

c) Try to clean the data yourself. There are many possible ways to deal with these "bad" timepoints, so you can choose one to investigate. Some options:

- Remove the timepoints completely.
- Replace the values at these time points with the mean of the good timepoints.
- Replace the values at these time points with a randomly selected value from the good timepoints.
- Another strategy of your choosing.

d) Explain which strategy you chose and why. These are decisions that scientists and engineers have to make. There are pros and cons of each. It's really important to consider these and document the analysis decisions that you make.

e) Discuss with your partner/table or reflect on your own: what are some of the potential ethical implications of "cleaning" data?

f) Plot the activity in all of the brain regions over time for your "cleaned" data.

g) Plot the correlation matrix figure for your "cleaned" data.

h) What similarities and differences do you observe between the correlation matrix for the "unclean" and "clean" data? How do you interpret these differences? What do you take away from this for future work?

9.6 AI and Society Discussion

Framing (5 minutes)

Today we'll be talking about a constellation of issues that arise when AI technology, like facial recognition, is deployed in society. As the historian Melvin Kranzberg famously remarked, "Technology is neither good nor bad; nor is it neutral." As you saw in the reading from the night assignment, the effect of AI technology in society intersects a number of sensitive issues around race, class, and gender. Due to intersection of AI and these sensitive issues, it helps to take a few minutes to consider some guidelines for having fruitful discussions at your tables.

- Check out this poster put together by some Oliners with suggestions for having conversations on sensitive topics.
- The readings provide common information and framing, which we find is very helpful to finding common ground when discussing issues that individuals may relate to in very different ways.
- As you may be relatively new to these ideas, consider adopting a mindset of identifying key questions rather than necessarily coming to conclusions.
- When talking about the effect of a technology on a group that has been historically oppressed, you should be particularly sensitive in these discussions if you are not a member of this group. Be conscious of the ways in which your words might be experienced by those who may have faced a history of discrimination due to being a member of this group.

Unpacking the Readings (15 minutes)

Write down key concepts and clear up points of confusion on the readings.

- Joy Buolamwini's written testimony on bias in facial recognition technology (you may have watched this instead).
- Principles for Accountable Algorithms
- Google's Inclusive ML

Share Your Positive Application of AI (10 minutes)

Go around and share the application of AI that you think has the potential for great positive impact on society. Say a little bit about what you learned and how you think it would have a positive impact (e.g., in what ways and for whom).

What Did You Take Away? (15 minutes)

Please have one person take notes on this in some electronic format so they can submit it as part of their day survey

As a table, discuss what you took away from the readings and your discussion thus far. Here are some dimensions that you might want to explore.

1. What parts or quotes from the readings were most surprising / impactful to you?
2. Were you surprised by your reaction to reading any of the material (e.g., felt unexpectedly angry, sad, indifferent)?
3. What are the big questions that have been raised for you (these could be things that were already on your radar or new ones entirely)? These questions could relate to our society as a whole, your role as a citizien within society, your role as an Olin student, your future career path, etc.).
4. How do these readings intersect with knowledge you've gained from other contexts (e.g., in other courses or in your daily life experience)?

Solution 9.1

1. The column is a two-dimensional vector. The span is a line (slope = 1) in 2D space.
2. The columns are linearly-independent three-dimensional vectors. Their span is therefore a plane in 3D space. Since all the z-entries are zero, the plane is actually the xy-plane.
3. The columns are linearly-independent three-dimensional vectors. Their span is therefore a plane in 3D space. The plane is defined by the column vectors.

Solution 9.2

1. The Range of $\mathbf{A}$ is all multiples of $\begin{bmatrix}1\\1\end{bmatrix}$. Since $\mathbf{b}$ is a multiple of this vector then there is a solution. From an equation point of view, the solution is simply $x = 5$.
2. The Range of $\mathbf{A}$ is all multiples of $\begin{bmatrix}1\\1\end{bmatrix}$. Since $\mathbf{b}$ is not a multiple of this vector then there is no solution. From an equation point of view this makes sense because we are demanding that $x = 2$ and $x = 3$ at the same time.
3. The Range of $\mathbf{A}$ is a plane in 3D. Since $\mathbf{b}$ is the sum of the columns it must be in the Range of $\mathbf{A}$ and so there is a solution. From an equation point of view there are two linearly-independent equations in two unknowns.
4. The Range of $\mathbf{A}$ is a plane in 3D. Since $\mathbf{b}$ is not in this plane there is no solution. From an equation point of view this makes sense because trying to solve the equations results in an inconsistency.

Solution 9.3

In each case the "closest" point is the location where the "pointing" vector meets the surface at right angles, i.e. they are orthogonal.

Solution 9.4

1. Consider the linear system $\mathbf{Ax} = \mathbf{b}$ where $\mathbf{A} = \begin{bmatrix}1\\1\end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix}2\\3\end{bmatrix}$. (You've already thought about this earlier).
   a) Sketch the Range of $\mathbf{A}$ and locate the point in the Range that is closest to $\mathbf{b}$. (The Range is a straight line and the point is the orthogonal projection onto this line.)
   b) Multiply both sides of $\mathbf{Ax} = \mathbf{b}$ by $\mathbf{A}^T$ and solve the resulting linear system. (You should find that $x = 5/2$).
2. Consider the linear system $\mathbf{Ax} = \mathbf{b}$ where $\mathbf{A} = \begin{bmatrix}1 & 2\\3 & 4\\5 & 6\end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix}3\\7\\5\end{bmatrix}$. (You've already thought about this earlier).

a) Sketch the Range of $\mathbf{A}$ and locate the point in the Range that is closest to $\mathbf{b}$. (The Range is a plane in 3D and the point is the orthogonal projection onto this line.)

b) Multiply both sides of $\mathbf{Ax} = \mathbf{b}$ by $\mathbf{A}^T$ and solve the resulting linear system. (You should find that $x = -3$ and $y = 7/2$.)

Chapter 10

Night 5

Learning Objectives

Concepts

- Describe the physical significance of the mean and standard deviation of a data set.
- Describe the physical significance of correlation, anti-correlation or non-correlation of two variables.
- Approximate the mean and standard deviation from a histogram of the data.
- Interpret the meaning of a pair of images that has a Pearson Correlation Coefficient of: about 0; or 0.5; or 0.9.
- Interpret the physical/mathematical meaning of the diagonal and off-diagonal elements in a 2 x 2 correlation matrix, $\mathbf{C} = \mathbf{A}^T\mathbf{A}$, if given the equation for the Pearson Correlation Coefficient.

MATLAB skills

- Compute the dot product of two vectors
- Set up the appropriate matrices to compute the correlation coefficient between two variables.

10.1 Correlation

Now let's consider that we measure two different associated quantities and want to test whether these are linearly correlated (if one goes up, the other also goes up), anti-correlated (if one goes up the other goes down) or uncorrelated (the behavior of one cannot be predicted by watching the behavior of the other). (Please note that correlation has nothing to do with causality!). There are many different measures of correlation, but we will discuss here one of the most common, the Pearson Correlation Coefficient.

For a pair of associated datasets $X = \{x_i\}$ and $Y = \{y_i\}$, each with n elements, we define the Pearson Correlation Coefficient to be:

$$\rho(X, Y) = \frac{\sum_{i=1}^{n}(x_i - \mu_x)(y_i - \mu_y)}{(N-1)\sigma_x\sigma_y} \tag{10.1}$$

where μ_x, μ_y, σ_x and σ_y are the means and standard deviations of the datasets. Essentially, for each pair of values, we take the product of the variations from the mean, then sum these products up over all pairs of values and normalize by the expected variation as characterized by the standard deviation. If the two values are consistently always on the same side of the mean, then each term in the sum will contribute positively, and the total value will be close to one, indicating positive correlation. If the two values are consistently on the opposite sides of the mean, then each term in the sum will contribute negatives, and the total value will be close to negative one, indicating anticorrelation. If, for every pair, it is just as likely that the two values

will be on opposite sides of the mean as on the same side of the mean, then the sum will go to zero, and the two values are uncorrelated.

Consider the following data:

	A	B	C	D	E	F	G	H	I	J
3		Poverty	Infant Mort	White	Crime	Doctors	Traf Deaths	University	Unemployed	Income
4	Alabama	15.7	9.0	71.0	448	218.2	1.81	22.0	5.0	42,666
5	Alaska	8.4	6.9	70.6	661	228.5	1.63	27.3	6.7	68,460
6	Arizona	14.7	6.4	86.5	483	209.7	1.69	25.1	5.5	50,958
7	Arkansas	17.3	8.5	80.8	529	203.4	1.96	18.8	5.1	38,815
8	California	13.3	5.0	76.6	523	268.7	1.21	29.6	7.2	61,021
9	Colorado	11.4	5.7	89.7	348	259.7	1.14	35.6	4.9	56,993
10	Connecticut	9.3	6.2	84.3	256	376.4	0.86	35.6	5.7	68,595
11	Delaware	10.0	8.3	74.3	689	250.9	1.23	27.5	4.8	57,989
12	Florida	13.2	7.3	79.8	723	247.9	1.56	25.8	6.2	47,778
13	Georgia	14.7	8.1	65.4	493	217.4	1.46	27.5	6.2	50,861
14	Hawaii	9.1	5.6	29.7	273	317.0	1.33	29.1	3.9	67,214
15	Idaho	12.6	6.8	94.6	239	168.8	1.60	24.0	4.9	47,576

Exercise 10.1

1. Look over the data. By eye, which columns look correlated? Anticorrelated? Uncorrelated?
2. Choose your two favorite columns of data from this dataset. Input these into vectors in Matlab. For each of these vectors, subtract off the mean, and then divide out the standard deviation.
3. With these vectors, how would you directly compute the correlation coefficient between them? Go ahead and do this in MATLAB, and reflect on your result.

A note of warning. Correlation does not imply causation.

Exercise 10.2

To drive this point home, visit the Spurious Correlation Website. Follow the link at the bottom of the site to discover and plot a spurious correlation of your very own.

10.2 *Correlation: The Idea, the Math, and the MATLAB*

Here we are going to consider a matrix like the one you constructed above, based on the datasets, which has elements of the correlation coefficients. Let's first consider a matrix $\mathbf{B}$ which has the datasets as columns:

$$\mathbf{B} = \begin{pmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ \dots & \dots \end{pmatrix}$$

If we subtract out the means and divide out the standard deviation and a factor of square root of $N-1$, we get the matrix $\mathbf{A}$:

$$\mathbf{A} = \frac{1}{\sqrt{N-1}} \begin{pmatrix} \frac{x_1-\mu_x}{\sigma_x} & \frac{y_1-\mu_y}{\sigma_y} \\ \frac{x_2-\mu_x}{\sigma_x} & \frac{y_2-\mu_y}{\sigma_y} \\ \frac{x_3-\mu_x}{\sigma_x} & \frac{y_3-\mu_y}{\sigma_y} \\ \dots & \dots \end{pmatrix}$$

where μ_x, μ_y and σ_x, σ_y are the mean and standard deviations of each column, and N is the number of samples (rows). Now we can write the *correlation matrix* as $\mathbf{C} = \mathbf{A}^T\mathbf{A}$ which has elements of the self and cross correlations between the datasets.

Exercise 10.3

1. Before we start to use this idea, let's think through it a bit...
   a) What is the size of the matrix $\mathbf{C}$?
   b) What are the elements on the *diagonal* of this matrix?
   c) What are the elements of the off-diagonal? What is element C_{12} of this matrix? What is element C_{21}? What do you notice? Is this always going to be true? What about if we had three datasets? What can you say about the elements C_{13} vs C_{31}?
   d) If you create a data matrix that has completely identical columns of data, what should the correlation matrix look like?
   e) If you create a data matrix that has completely uncorrelated datasets, what should the correlation matrix look like?
2. Now let's actually implement this. For the sake of getting you more comfortable manipulating matrices in MATLAB, do the following at the command line, and see if you can predict what each term will do before you type it in. For each command, explain what MATLAB is doing.

```
>> a = [(1:10)',rand(10,1)]
>> o = ones(size(a,1),1)
>> m = o*mean(a)
>> s = o*std(a)
>> b = (a-m)./s
>> c=(1/(size(a,1)-1))*b'*b
```

3. What would happen if you used this matrix for a instead:

```
>> a = [(1:10)',(1:10)'/2 -(1:10)' ]
```

   Make a prediction, and try it out.
4. Now learn how to use the MATLAB function `corrcoef`, which will compute the correlation coefficients using the original matrix.

10.3 From Data to Dimensions

Thus far we've made a distinction between vectors as representing points in space, and vectors as representing data (e.g., a list of intensity values for pixels). But if you wanted to, there's no reason you couldn't think of a vector of n data points as representing a point in n dimensional space (and, in fact, it would be very powerful *to*). If you did this, you could define all kinds of interesting things. For example, you could ask about the magnitude of the vector (i.e., the size of a data point), the distance from one vector (i.e., data point) to another, the dot product of two vectors (i.e., data points), etc.

Exercise 10.4

1. As a thought experiment, think about the following questions.
   a) Imagine you have grayscale images that are 100×100 pixels. If you represent each image as a vector, how large is the vector space?
   b) If the intensity of pixel ij is a_{ij}, come up with an expression for the magnitude of the vector that describes the image $\mathbf{a}$.
   c) Come up with an expression that represents the distance between images $\mathbf{a}$ and $\mathbf{b}$.
   d) What does $\mathbf{a}^T\mathbf{b}$ tell you? What vector operation is this?
2. Now that you've thought through this about, load three face images, and calculate the distance between each pair (i.e., take the difference between the two images, square each element, then sum up all the squared elements and take the square root of the sum). Do your answers make sense?

10.4 Correlation in Facial Recognition

Kinds of Correlation in an Image Set

If we think about pictures now, we can think about two different correlations: the correlation between a given pair of pixels (across all the pictures in a data set), and the correlation between pictures (across all the pixels in those images). In order to compute an accurate correlation coefficient, you need to have multiple data points in each set being correlated, e.g., many pixels in each picture being correlated, or many pictures across which a pair of pixels (pixel locations) can be correlated.

Think about what each of these correlations *means*. What would a high correlation between a given pair of images mean? What about a high correlation between a given pair of pixels (e.g., the upper-left-most

pixel and the upper-right-most pixel)? It might help to open a few face images or draw some face sketches to think about.

Exercise 10.5

Consider six grayscale pictures, each with a resolution of m x n pixels.

1. What is the size of the data matrix containing these six pictures as the columns?
2. What is the expression for the correlation matrix between the pictures? What size is this correlation matrix?
3. What is the expression for the correlation matrix between different pixels? Pay careful attention to the mean and standard deviation you are using. What is the size of this correlation matrix?
4. People's faces are approximately left-right symmetric. How would you expect this to affect the entries in the correlation matrix between different pixels?

Test your understanding...

- Pull in **six images** from the class data matrix (`test_images` variable in `face_bases.mat` file) and put them in a variable called `faces`. Each of these images should come from different people - there are 8 images per person stored in the data matrix.
- Use the `resample` command to bring them down to a smaller resolution (e.g., 25 x 25) using
  `dfaces = imresize(faces,[25 25]);`
  his should be a 25 x 25 x 6 matrix.
- Now reshape them appropriately to create a matrix in which each column is a (reshaped) face using
  `rdfaces = reshape(dfaces,size(dfaces,1) * size(dfaces,2),size(dfaces,3));`

Exercise 10.6

1. Find the correlation between six different images. Which images have the highest correlation?
2. Now find the correlation between pixels across images. Try taking a single column of this matrix and reshape that column into an image. What does that image tell you? You may want to repeat this reshape and visualization for columns 1, 25, 400, and 625 to get a feel for what is happening.

10.5 Diagnostic Quiz

Please see Canvas for the questions.

Solution 10.3

1. a) Since $\mathbf{A}$ has size $N \times 2$, we know $\mathbf{A}^T$ has size $2 \times N$. Then $\mathbf{C} = \mathbf{A}^T\mathbf{A}$ has size 2×2.
   b) Each element of the diagonal will be 1.
   c) The elements $\mathbf{C}_{12} = \mathbf{C}_{21}$ is the correlation between the two columns of data. Regardless of size, the correlation matrix will be symmetric.
   d) If the data matrix had identical columns of data, the correction matrix would be all 1s.
   e) If the data is uncorrelated, the off-diagonal entries in the correlation matrix will be all 0s, so it will be the identity matrix. (Note: real data is unlikely to have 0 correlation, just by accident, so it will just have numbers that are close to 0.)
2. You should end up with a correlation matrix between the two columns of `a`.
3. Again, you end up with a correlation matrix between the two columns of `a`, but since the numbers are non-random and negatively correlated, you get the matrix $\mathbf{c} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$.
4. You should use "help" or "doc" to find information on how to use this function.

Solution 10.4

1. Each pixel is represented by a number. So the vector representing an image has 10000 entries and thus lives in 10000-dimensional space.
2. The magnitude of the vector is given by
$$\|\mathbf{a}\| = \left(\sum_{i,j} a_{i,j}^2 \right)^{1/2}.$$
3. The distance between two images $\mathbf{a}$ and $\mathbf{b}$ is given by
$$\|\mathbf{a} - \mathbf{b}\| = \left(\sum_{i,j} (a_{i,j} - b_{i,j})^2 \right)^{1/2}.$$
4. The dot product $\mathbf{a}^T\mathbf{b}$ tells you how similar the two vectors are.

Solution 10.5

1. Each picture is represented by mn data points. So the data matrix containing these six pictures as columns is $mn \times 6$.
2. To find the correlation we can either use the MATLAB code from Exercise 10.3 or the command `corrcoef`. The correlation matrix will be 6×6.

3. To find the correlation between pixels, we need to take the transpose of our data matrix. This new data matrix will be $6 \times mn$, since we have mn variables (each pixel) and 6 observations (within in picture). The correlation matrix will then be $mn \times mn$.

4. High correlation between pixels equidistant from centerline.

Solution 10.6

1. To find the correlation matrix between the six images, enter
   `» corrcoef(rdfaces)`.

2. To find the correlation across pixels, enter
   `» pixels=corrcoef(transpose(rdfaces));`. We can reshape the first column of this matrix and convert it into an image using
   ```
   » pixels1=reshape(pixels(:,1),25,25);
   » imagesc(pixels1).
   ```
   The (i, j) entry of this image tells us how similar the top-left pixel is to the (i, j) pixel. (Note: the pixels are "numbered" 1–625 going down the first column, then down the second column, and so on.) Repeating this reshape and visualization procedure on column 400 will give you the correlation between pixel 400 and each of the other pixels, for example.

Chapter 11

Day 6: AI Discussion, Smile Detection and Eigenthings

11.1 Schedule

- 0900-0915: Debrief
- 0915-1000: AI and Society Discussion
- 1000-1030: Smile Detection—Concepts
- 1030-1045: Coffee
- 1045-1115: Machine Learning
- 1115-1145: Smile Detection—Implementation
- 1145-1210: Eigenthings
- 1210-1230: Review/preview

11.2 Debrief and Synthesis [15 mins]

- Please discuss your overnight work with your table-mates, and get help with the ideas that you are still confused by.

11.3 AI and Society Discussion [45 mins]

Framing (5 minutes)

Today we'll be talking about a constellation of issues that arise when AI technology, like facial recognition, is deployed in society. As the historian Melvin Kranzberg famously remarked, "Technology is neither good nor bad; nor is it neutral." As you saw in the reading from the night assignment, the effect of AI technology in society intersects a number of sensitive issues around race, class, and gender. Due to intersection of AI and these sensitive issues, it helps to take a few minutes to consider some guidelines for having fruitful discussions at your tables.

- Check out this poster put together by some Oliners with suggestions for having conversations on sensitive topics.
- The readings provide common information and framing, which we find is very helpful to finding common ground when discussing issues that individuals may relate to in very different ways.
- As you may be relatively new to these ideas, consider adopting a mindset of identifying key questions rather than necessarily coming to conclusions.
- When talking about the effect of a technology on a group that has been historically oppressed, you should be particularly sensitive in these discussions if you are not a member of this group. Be conscious of the ways in which your words might be experienced by those who may have faced a history of discrimination due to being a member of this group.

Unpacking the Readings (15 minutes)

Write down key concepts and clear up points of confusion on the readings.

- Joy Buolamwini's written testimony on bias in facial recognition technology (you may have watched this instead).
- Principles for Accountable Algorithms
- Google's Inclusive ML

Share Your Positive Application of AI (10 minutes)

Go around and share the application of AI that you think has the potential for great positive impact on society. Say a little bit about what you learned and how you think it would have a positive impact (e.g., in what ways and for whom).

What Did You Take Away? (15 minutes)

Please have one person take notes on this in some electronic format so they can submit it as part of their day survey

As a table, discuss what you took away from the readings and your discussion thus far. Here are some dimensions that you might want to explore.

1. What parts or quotes from the readings were most surprising / impactful to you?
2. Were you surprised by your reaction to reading any of the material (e.g., felt unexpectedly angry, sad, indifferent)?
3. What are the big questions that have been raised for you (these could be things that were already on your radar or new ones entirely)? These questions could relate to our society as a whole, your role as a citizen within society, your role as an Olin student, your future career path, etc.).
4. How do these readings intersect with knowledge you've gained from other contexts (e.g., in other courses or in your daily life experience)?

11.4 Smile Detection—Concepts [30 mins]

In this session we are going to use our toolbox of linear algebra skills to "detect" whether or not a person is smiling in a photograph. The approach that we will take is very common in **machine learning** - we will use a dataset to **train** our algorithm, and we will use a different dataset to **test** our algorithm. We will first develop the conceptual framework and then implement the approach in MATLAB.

The Big Idea

Let's assume that we have 100 training photos of faces, each consisting of a 5 by 5 grid of pixels. Let's pack these into a matrix $\mathbf{A}$ with 100 rows and 25 columns, i.e. every row is a different face and every column is a different pixel.

Let's also assume that we have already classified every training face as "smiling" or "not-smiling". Let's create a column vector $\mathbf{b}$ with 100 rows (corresponding to each face) which has either 1 (smiling) or 0 (not-smiling).

Let's now develop a linear system of algebraic equations by trying to express the vector $\mathbf{b}$ as a linear combination of the columns of $\mathbf{A}$, i.e.

$$\mathbf{Ax} = \mathbf{b}$$

Notice that the vector $\mathbf{x}$ is a column vector with 25 rows - one row for each pixel. Since there are more rows than columns we know that an **exact** solution does not exist, so we will find the **approximate** solution by orthogonal projection, i.e. we will solve

$$\mathbf{A}^T\mathbf{Ax} = \mathbf{A}^T\mathbf{b}$$

Now that we have the vector $\mathbf{x}$, let's use it to detect whether a test image is smiling. Assuming that the test image is packed into a single row vector $\mathbf{t}$ (with 25 columns) then the product

$$\mathbf{tx}$$

will return a scalar. If this scalar is close to "1" then we predict the face is smiling. If this scalar is close to "0" then we predict the face is not smiling.

Exercise 11.1

In this exercise you will be carefully reading and interpreting this big idea. We are including these questions as a scaffold, pointing out interesting features along the way.

1. Read "The Big Idea" again!
2. Interpret what it means to write down the linear system of equations $\mathbf{Ax} = \mathbf{b}$ and give a meaning to the vector $\mathbf{x}$.
3. Interpret the product $\mathbf{A}^T\mathbf{A}$ and the product $\mathbf{A}^T\mathbf{b}$.
4. The vector $\mathbf{x}$ does not satisfy $\mathbf{Ax} = \mathbf{b}$ exactly. What does the expression $\mathbf{Ax} - \mathbf{b}$ tell you?
5. How would you decide whether your "trained" algorithm was worth using on a test dataset?
6. Assume you had 40 test images with 25 pixels each and that you pack them into a matrix $\mathbf{T}$ with 40 rows and 25 columns. Write down the matrix-vector product you would use for smile detection on this test dataset.
7. How would you measure the accuracy of your predictions if we also provided you with the data on whether each test image was smiling or not?

11.5 Machine Learning in general [30 mins]

Machine learning is an interdisciplinary field concerned with the idea that rather than preprogramming machines to solve tasks explicitly, instead we can program them to learn to solve tasks through experience. In order to connect this idea to what you've done thus far, first we'll introduce a somewhat more formal definition of machine learning.

> *A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E.*
>
> — *Tom Mitchell*

Let's take the smile detection problem that you just framed in the preceding section of the document.

Symbol from Mitchell's Definition	Smile Detection
E	5 by 5 grids of pixels with corresponding labels as to whether or not the person in the grid is smiling
T	Given a new 5 by 5 grid of pixels, predicting whether the person in the image is smiling.
P	Accuracy on predicting whether a person is smiling (e.g., percent correct)

The last part of the definition states that in order to say a program is "learning", it should "improve with experience E." In the case of smile detection, what this means is that given more images with corresponding indication of whether each person is smiling, our program should get better at predicting whether a face is smiling. Later in this document when you implement smile detection, you'll see that the framing of smile detection as an LSAE problem absolutely meets this definition (and does a surprisingly good job at it too)!

Exercise 11.2

In this exercise you and your table-mates will be working to frame various machine learning problems as systems of linear equations. In doing so, you should answer the following questions.

- What is the thing you want to predict (i.e., what is your $\mathbf{b}$ in $\mathbf{Ax} = \mathbf{b}$).
- What quantities are you using to make your prediction (i.e., what do each of the rows and columns in the matrix $\mathbf{A}$ represent)?
- While $\mathbf{x}$ would be determined by solving $\mathbf{Ax} = \mathbf{b}$, come up with a guess as to what $\mathbf{x}$ would be (don't worry about coming up with numbers. Instead, try to identify the sign of each element of $\mathbf{x}$ and whether its magnitude is large or small relative to the other entries of $\mathbf{x}$).
- How would you measure how well your system works? For example, for smile detection we might apply our learned model $\mathbf{x}$ to new data and see how often it correctly predicts the facial expression (smiling vs. not smiling) of the person in each image.

- What sorts of issues of bias might you have to worry about in this system?

1. AirBnB has a smart pricing option that lets folks who list their properties on the site have the price for those listings determined automatically. How might you frame the creation of this smart pricing tool as solving a system of linear equations? We suggest that you follow the steps outlined at the start of this exercise. You might want to do a quick AirBnB search if you are not familiar with the site.
2. Netflix suggests content to a user that they are likely to enjoy based on their viewing history as well as the viewing histories of others. In the past (although we think possibly not anymore), they used to also incorporate ratings data (1 to 5 stars) of particular movies from both the user receiving the recommendation and other users on the site. How might you frame the creation of this recommendation system in terms of solving a system of linear equations? We suggest that you follow the steps outlined at the start of this exercise. You might want to do a quick Netflix search if you are not familiar with the site.

11.6 Smile Detection—Implementation [30 mins]

Please download the file `smiles.mat` from the canvas site. If you load this file in MATLAB, you will then have access to the following variables in your workspace.

```
train_data       - a 3D array containing 19685 24 x 24 pixel
                   images of faces
smile_flag_train - a vector of the same length as the number
                   of images in train_data, with 1s indicating
                   which images are smiling
test_data        - 500 24 x 24 pixel images of faces
smile_flag_test  - a vector of the same length as the number
                   of images in test_data, with 1s indicating
                   which images are smiling
```

The 'train data' and the associated 'smile flag train' are the sets of data you should use to develop your mathematical model. The 'test data' and its associated 'smile flag test' are the sets of data you should use to test your algorithm when you are finished!

Exercise 11.3

For this exercise we recommend that you use our walkthrough notebook. The notebook has embedded solutions or you can try it with minimal scaffolding using the suggested process below. Even if you decide not to use the walkthrough notebook, it's worth running the embedded solutions to pickup some techniques for visualizing your smile detector model. In any case, as you move into actually implementing the smile detector, we recommend you work as pairs at your table. Working in a pair will allow you to have someone to bounce ideas off of and also make sure you can see the

laptop easily.
Only if you decide NOT to use the walkthrough notebook, you should consider following the procedure below to implement the smile detector.

1. Sketch out a set of steps you would take in order to implement smile detection. (Just words here - no code. e.g. we will have to pack all images into a single matrix)
2. Turn this set of steps into MATLAB pseudo-code. Identify important coding elements without implementing, e.g. we will use **reshape** to pack the given dataset into a matrix.
3. Review the documentation for MATLAB functions that will be used and be clear on how to use them before implementation, e.g. » help reshape
4. Methodically implement smile detection in MATLAB, testing as you go.

11.7 Introduction to Eigenthings [25 mins]

We are now going to learn the secret of the genie ...

Eigenvalues and Eigenvectors: Definition and Notation

Consider a square $n \times n$ matrix $\mathbf{A}$. A vector $\mathbf{v}$ is said to be an eigenvector of $\mathbf{A}$ with corresponding eigenvalue λ if $\mathbf{v}$ is not a vector of all zeros, and

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v}. \tag{11.1}$$

If we treat $\mathbf{A}$ as a transformation matrix then $\mathbf{v}$ is an eigenvector of $\mathbf{A}$ if it is simply scaled when acted on by the matrix $\mathbf{A}$. In other words, $\mathbf{v}$ does not change direction when acted upon by $\mathbf{A}$. In general, an $n \times n$ matrix has exactly n eigenvalues (although some of these may be repeated and some of these may be complex!). Note that any scalar multiple of an eigenvector of a matrix is also an eigenvector of that matrix - it's only the direction of the eigenvector that matters.

In the next overnight assignment we are going to develop formal techniques for finding the eigenvalues and eigenvectors of matrices. For now, we are going to focus on concepts and developing some intuition.

Exercise 11.4

1. Show that $\mathbf{v} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$ is an eigenvector of the following matrix by computing the product $\mathbf{A}\mathbf{v}$, and find the corresponding eigenvalue

$$A = \begin{bmatrix} 1 & 1 \\ 2 & 0 \end{bmatrix} \tag{11.2}$$

2. On the same axes, plot the vector representing $\mathbf{v} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$ and $\mathbf{Av}$. Does the plot confirm that this is an eigenvector?
3. On the same axes, plot the vector representing $\mathbf{u} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 1 \end{bmatrix}$ and $\mathbf{Au}$. Is this is an eigenvector of $\mathbf{A}$?

Eigenvalues and eigenvectors of a diagonal matrix

Recall from our earlier work that the matrix

$$\mathbf{A} = \begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}$$

scales vectors by a factor of 2 in the x-direction and by a factor of 3 in the y-direction. Thus a vector that had a non-zero component only in the x direction will be scaled by a factor of 2 when transformed by this matrix. In other words, $\lambda_1 = 2$ is an eigenvalue with corresponding eigenvector $\mathbf{v}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, and that $\lambda_2 = 3$ is an eigenvalue with corresponding eigenvector $\mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$. Let's check if the first one is true:

$$\mathbf{Av}_1 = \begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \end{bmatrix} = 2 \begin{bmatrix} 1 \\ 0 \end{bmatrix} = 2\mathbf{v}_1$$

Therefore $\lambda_1 = 2$ is an eigenvalue with corresponding eigenvector $\mathbf{v}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$.

Exercise 11.5

Confirm that $\lambda_2 = 3$ is an eigenvalue with corresponding eigenvector $\mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ by computing the product $\mathbf{Av}_2$.

Based on this example, we can heuristically guess that the eigenvalues of an $n \times n$ diagonal matrix are the entries on the diagonal. The n eigenvectors each have a single 1 in them, with the remaining entries being zero.

Exercise 11.6

What are the eigenvalues and eigenvectors of the following diagonal matrices

1. $$\mathbf{A} = \begin{bmatrix} -3 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 4 \end{bmatrix}$$

2. $$\mathbf{A} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Exercise 11.7

1. Assume that $\mathbf{v} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ is an eigenvector with eigenvalue 3. Construct an appropriate matrix A with this eigenvalue and eigenvector by first rotating $\mathbf{v}$ onto the x-axis, scaling it by 3, and then rotating back.

Exercise 11.8

What is one eigenvector of the following matrix?

$$\mathbf{R} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{11.6}$$

11.8 Review and Preview [20 minutes]

Solution 11.1

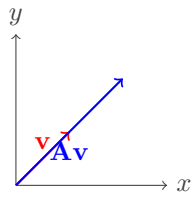
1. Read, read, read ....
2. We are trying to take a linear combination of the data in order to predict whether each image is smiling or not. The vector $\mathbf{x}$ is the magic set of weights we have to use. Its size is the same as the number of pixels, so maybe it should look like a mask that we can place over an image to tell us whether it is smiling.
3. The product $\mathbf{A}^T\mathbf{A}$ is like a pixel to pixel correlation matrix, except we haven't scaled the data matrix $\mathbf{A}$. The product $\mathbf{A}^T\mathbf{b}$ is the sum of the images that are smiling.
4. The expression $\mathbf{Ax} - \mathbf{b}$ tells us the error in predicting whether a training image is smiling or not.
5. We could add up how often the predictor is correct and divide by the number of images to get an estimate of the accuracy. We would decide on a cut-off before we used it on a test dataset.
6. It is simply $\mathbf{Tx}$.
7. As before. Determine how many we got correct and average it.

Solution 11.3

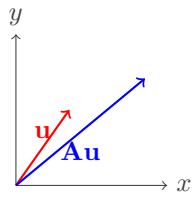
You can use the solutions that are embedded in the walkthrough notebook.

Solution 11.4

1. Compute that
$$A\mathbf{v} = \begin{bmatrix}\sqrt{2}\\ \sqrt{2}\end{bmatrix} = 2\mathbf{v}$$
and so the corresponding eigenvalue is $\lambda = 2$.
2. As we can see in the picture below, both $\mathbf{v}$ and $\mathbf{Av}$ point in the same direction, which confirms $\mathbf{v}$ is an eigenvalue of $\mathbf{A}$



3. As we can see in the picture below, $\mathbf{u}$ and $\mathbf{Au}$ point in different directions, so $\mathbf{u}$ is not an eigenvector of $\mathbf{A}$



Solution 11.5

We compute that

$$\mathbf{A}\mathbf{v}_2 = \begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 3 \end{bmatrix} = 3 \begin{bmatrix} 0 \\ 1 \end{bmatrix} = 3\mathbf{v}_2.$$

Solution 11.6

1. The eigenvalues are $\lambda_1 = -3$, $\lambda_2 = -1$ and $\lambda_3 = 4$ and the corresponding eigenvectors are $\mathbf{v}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$, $\mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ and $\mathbf{v}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$.
2. The eigenvalues are $\lambda_1 = 2$, $\lambda_2 = 4$ and $\lambda_3 = 0$ and the corresponding eigenvectors are $\mathbf{v}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$, $\mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ and $\mathbf{v}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$.

Solution 11.7

1. First rotate it using a 45 degree clockwise rotation matrix

$$\mathbf{R}(-45) = \begin{bmatrix} cosd(-45) & -sind(-45) \\ sind(-45) & cosd(-45) \end{bmatrix} \tag{11.3}$$

Now that it is along the x-axis we can scale it by 3 using the scaling matrix

$$S = \begin{bmatrix} 3 & 0 \\ 0 & 0 \end{bmatrix} \tag{11.4}$$

We then rotate it back using $\mathbf{R}(45)$. Multiplying these matrices together gives

$$\mathbf{A} = \begin{bmatrix} 1.5 & 1.5 \\ 1.5 & 1.5 \end{bmatrix} \tag{11.5}$$

Solution 11.8

The vector $\mathbf{v} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ is an eigenvector of $\mathbf{R}$ because it is the rotation axis and therefore remains unchanged on rotation.

Chapter 12

Night 6: Eigenvalues and Eigenvectors

Learning Objectives

Concepts

- Compute the eigenvalues and eigenvectors of a 2×2 matrix by hand
- Compute the eigenvalues and eigenvectors of an $n \times n$ matrix using MATLAB
- Describe the geometric meaning of eigenvalues and eigenvectors
- Use eigenvectors to compute and interpret directions of variation in data

MATLAB skills

- Compute the eigenvectors and eigenvalues of a given matrix
- From a given dataset, set up the relevant matrices and compute the covariance matrix of the dataset.

What is this about?

The big ideas of this assignment are eigenvectors and eigenvalues. Recall that when you multiply a vector by a matrix, the resulting vector usually points in a different direction. An eigenvector of a square matrix is a vector which does not change direction when multiplied by that matrix. It can only change in length. The eigenvalue corresponding to this eigenvector is the scale factor that is applied to that eigenvector as a result of the matrix multiplication. Therefore, the eigenvector of a matrix points in a special direction – its a direction that is not modified by the linear transformation associated with that matrix. This is an idea that we will keep coming back to in a number of different ways throughout QEA (including next semester). The ideas contained here can be applied in many ways (many of which we won't get to until next semester) such as

- Directions of greatest variation in data.
- Natural co-ordinates of systems.
- Frequency response of filters.
- Analysis of dynamical systems.

Reference Material

- Eigenvalues and Eigenvectors by 3Blue1Brown (watch first 14 mins)
- Paul's Online Notes. Review : Eigenvalues and Eigenvectors
- Intro to eigenvectors by PatrickJMT

- Calculating eigenvalues and eigenvectors of a 2×2 matrix by PatrickJMT.

12.1 Calculating Eigenvalues and Eigenvectors of Matrices

Recall from class that λ is an eigenvalue of a matrix $\mathbf{A}$ with corresponding eigenvector $\mathbf{v}$ if $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$. Geometrically, this means that the matrix $\mathbf{A}$ doesn't change the direction of $\mathbf{v}$, it simply scales it by a factor of λ.

Given a square matrix, how can we find its eigenvalues and eigenvectors? In class, we calculated these by hand for the special case of diagonal matrices, and now we will move to generic 2×2 matrices. For general square matrices which are larger than 2×2, we will use MATLAB's `eig` to compute the eigenvalues and eigenvectors.

Finding eigenvalues

So far we've dealt with matrices for which it is possible to think your way to the eigenvalues. For general matrices, this is rarely the case, and we need a method that is foolproof. The method most widely adopted involves the determination of an algebraic equation for the eigenvalues, usually known as the *characteristic* equation. For this reason, eigenvalues are often known as characteristic values.

Let's start with an example. Consider the matrix

$$\mathbf{A} = \begin{bmatrix} 18 & -2 \\ 12 & 7 \end{bmatrix}$$

The definition of an eigenvalue and eigenvector imply that we are seeking λ and $\mathbf{v}$ which satisfy

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v}.$$

We subtract $\lambda\mathbf{v}$ from both sides

$$\mathbf{A}\mathbf{v} - \lambda\mathbf{v} = \mathbf{0}$$

and then factor the left hand side to give

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{v} = \mathbf{0}$$

For this example we have

$$\mathbf{A} - \lambda\mathbf{I} = \begin{bmatrix} 18 - \lambda & -2 \\ 12 & 7 - \lambda \end{bmatrix}.$$

We are only interested in $\mathbf{v}$ that are nonzero, i.e., $\mathbf{v}$ is not the vector of all zeroes. (This is because $\mathbf{v} = \mathbf{0}$ is always a solution to $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$ for any $\mathbf{A}$ and any λ, so it's not very interesting or informative.) Assuming $\mathbf{v}$ is nonzero implies that the matrix $(\mathbf{A} - \lambda\mathbf{I})$ is not invertible. Why? If $(A - \lambda\mathbf{I})$ were invertible, then we could rearrange the equation to get

$$\mathbf{v} = (\mathbf{A} - \lambda\mathbf{I})^{-1}\mathbf{0} = \mathbf{0}$$

which contradicts our assumption that $\mathbf{v} \neq \mathbf{0}$. THerefore, $(\mathbf{A} - \lambda\mathbf{I})$ is not invertible.

Since $(\mathbf{A} - \lambda\mathbf{I})$ is not invertible, it must have determinant zero. In other words,

$$\det(\mathbf{A} - \lambda\mathbf{I}) = 0.$$

In our example, this implies that

$$\det(\mathbf{A} - \lambda\mathbf{I}) = (18 - \lambda)(7 - \lambda) + 24 = 0.$$

This is called the characteristic equation:

$$(18 - \lambda)(7 - \lambda) + 24 = 0$$

or, rearranged,

$$\lambda^2 - 25\lambda + 150 = 0.$$

The characteristic equation is a polynomial with the variable λ that arises by setting the determinant of $(\mathbf{A} - \lambda\mathbf{I})$ equal to zero. The solutions to this polynomial give the eigenvalues λ. In our example, the polynomial can be factored

$$(\lambda - 15)(\lambda - 10) = 0$$

so that gives eigenvalues $\lambda_1 = 10$ and $\lambda_2 = 15$. (We could use the quadratic formula if necessary.)

Let's retrace our steps: If λ is either 10 or 15, then the determinant of $(\mathbf{A} - \lambda\mathbf{I})$ is zero. This implies that $(\mathbf{A} - \lambda\mathbf{I})$ is not invertible, so we can look for nonzero solutions $\mathbf{v}$ to $(\mathbf{A} - \lambda\mathbf{I})\mathbf{v} = \mathbf{0}$ and those $\mathbf{v}$ are eigenvectors associated to the eigenvalue λ.

In summary, here's the general procedure for finding the eigenvalues of a matrix:

1. Rearrange $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$ to get $(\mathbf{A} - \lambda\mathbf{I}) = \mathbf{0}$.
2. Compute the determinant of $(\mathbf{A} - \lambda\mathbf{I})$.
3. Since the matrix is not invertible, we set that determinant equal to zero: $\det(\mathbf{A} - \lambda\mathbf{I}) = 0$. This gives a polynomial in λ, known as the characteristic equation.
4. Solve the polynomial for the roots λ. Those are the eigenvalues.

Exercise 12.1

1. You already know that the eigenvalues of a diagonal matrix are just the entries on the diagonal. Using the above procedure, confirm that

   $$\mathbf{A} = \begin{bmatrix} 2 & 0 \\ 0 & -3 \end{bmatrix}$$

   has eigenvalues $\lambda_1 = 2$ and $\lambda_2 = -3$.

2. Notice that one of the eigenvalues is positive and one is negative. The eigenvector associated with $\lambda_1 = 2$ is $\mathbf{v}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and the eigenvector associated with $\lambda_2 = -3$ is $\mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$. Plot $\mathbf{v}_1, \mathbf{v}_2, \mathbf{A}\mathbf{v}_1$ and $\mathbf{A}\mathbf{v}_2$. What affect does the negative sign in the eigenvalue have? In other words, what is the difference between a negative and positive eigenvalue?

It's worth noting that eigenvalues come in more flavors than positive or negative. They can also be complex numbers. For now, will focus on matrices with real eigenvalues, but if you're curious about the complex case, you can learn about it in this worksheet (ignore the first page).

Eigenvalues in terms of the trace and determinant

The steps we took just in the previous sections will work for any matrix, so let's apply it to the most general 2×2 matrix

$$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

Again we seek λ and $\mathbf{v}$ so that

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$$

or equivalently

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{v} = \mathbf{0}$$

Non-zero solutions for $\mathbf{v}$ exist when

$$\det(\mathbf{A} - \lambda\mathbf{I}) = 0$$

so in the general case the characteristic equation is

$$(a - \lambda)(d - \lambda) - bc = 0$$

Expanding and simplifying gives

$$\lambda^2 - (a + d)\lambda + (ad - bc) = 0$$

which is a second-order polynomial in λ with two coefficients. Notice that the last one is just $det(\mathbf{A})$ and the middle one involves the sum of the diagonal entries of $\mathbf{A}$, which is known as the trace of $\mathbf{A}$ or $tr(\mathbf{A})$ for short. The characteristic equation is therefore

$$\lambda^2 - tr(\mathbf{A})\lambda + det(\mathbf{A}) = 0$$

Finally, let's consider the solutions of the characteristic equation for a 2x2 matrix. Using the quadratic formula we have

$$\lambda = \frac{tr(\mathbf{A}) \pm \sqrt{tr(\mathbf{A})^2 - 4det(\mathbf{A})}}{2}$$

If you recall all the work you did in school with the solutions to the quadratic, you will notice that there are two solutions as expected, one for each eigenvalue. Furthermore, the solutions may be *complex* if

$$tr(\mathbf{A})^2 - 4det(\mathbf{A}) < 0.$$

Exercise 12.2

Determine the trace and determinant of the following 2x2 matrices and then write down the

corresponding characteristic equation. Solve the characteristic equation to find the eigenvalues.

1. $$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}$$
2. $$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 2 \end{bmatrix}$$

Exercise 12.3

Optional problem if you're interested in further exploring the relationship between the trace, determinant, and eigenvalues.

1. Use the solutions of the characteristic equation to prove that $\lambda_1 + \lambda_2 = tr(\mathbf{A})$.
2. Use the solutions of the characteristic equation to prove that $\lambda_1 \lambda_2 = det(\mathbf{A})$.
3. Use the solutions of the characteristic equation to prove that the eigenvalues of a symmetric 2×2 matrix are real.

Finding Eigenvectors

In the example in the previous section, we discovered that the eigenvalues of

$$\mathbf{A} = \begin{bmatrix} 18 & -2 \\ 12 & 7 \end{bmatrix}$$

are $\lambda_1 = 10$ and $\lambda_2 = 15$. How do we find the corresponding eigenvectors $\mathbf{v}_1$ and $\mathbf{v}_2$?

First, let's find the eigenvector corresponding to $\lambda_1 = 10$. Remember that we knew λ_1 was an eigenvalue because it solved the characteristic equation, i.e., $\det(\mathbf{A} - \lambda_1 \mathbf{I}) = \mathbf{0}$. This is important because it implies $(\mathbf{A} - \lambda_1 \mathbf{I})$ is non-invertible, and therefore, there exists a nonzero vector $\mathbf{v}_1$ such that $(\mathbf{A} - \lambda_1 \mathbf{I})\mathbf{v}_1 = \mathbf{0}$. But it's not enough just to know that such a vector exists, we want to know exactly what it is.

In our running example, this means we are looking for $\mathbf{v}_1$ such that

$$(\mathbf{A} - \lambda_1 \mathbf{I})\mathbf{v}_1 = \begin{bmatrix} 18 - 10 & -2 \\ 12 & 7 - 10 \end{bmatrix} \mathbf{v}_1 = \begin{bmatrix} 8 & -2 \\ 12 & -3 \end{bmatrix} \mathbf{v}_1 = \mathbf{0}.$$

Let's write $\mathbf{v}_1$ in terms of its components

$$\mathbf{v}_1 = \begin{bmatrix} a \\ b \end{bmatrix},$$

to get the equation

$$\begin{bmatrix} 8 & -2 \\ 12 & -3 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

This gives us two equations

$$8a - 2b = 0 \text{ and } 12a - 3b = 0.$$

But these equations provide the same information: they both imply that $b = 4a$. This is because $(\mathbf{A} - \lambda_1 \mathbf{I})$ is not invertible, so the rows are linear dependent. The system of linear equations implied by $(\mathbf{A} - \lambda_1 \mathbf{I})\mathbf{v}_1 = \mathbf{0}$ has infinitely many solutions of the form $\begin{bmatrix} a \\ 4a \end{bmatrix}$ for any a. Letting $a = 1$, we get $\mathbf{v}_1 = \begin{bmatrix} 1 \\ 4 \end{bmatrix}$.

If we let $a = 5$, we would have the eigenvector $\begin{bmatrix} 5 \\ 20 \end{bmatrix}$. This hints at an important fact about eigenvectors: *we only care about an eigenvector's direction, not its length.* So we could have chosen $\mathbf{v}_1$ to be any vector pointing the same direction as $\begin{bmatrix} 1 \\ 4 \end{bmatrix}$ (such as $\begin{bmatrix} 5 \\ 20 \end{bmatrix}$ or $\begin{bmatrix} -2 \\ -8 \end{bmatrix}$). We often speak about "the" eigenvector corresponding to an eigenvalue, but only the direction of the eigenvector is unique, not the length.

Exercise 12.4

Using the basic eigenvalue/eigenvector equation

$$\mathbf{A}\mathbf{v} = \lambda \mathbf{v}$$

show that if $\mathbf{v}$ is an eigenvector for λ, then $c\mathbf{v}$ is also an eigenvector for λ, where c is any constant.

Exercise 12.5

We can always check that λ_1 and $\mathbf{v}_1$ are the corresponding eigenvalue and eigenvector for the matrix $\mathbf{A}$ by plugging them into the equation $\mathbf{A}\mathbf{v}_1 = \lambda_1 \mathbf{v}_1$ and verifying that it holds.

Use this procedure to check that $\lambda_1 = 10$ and $\mathbf{v}_1 = \begin{bmatrix} 1 \\ 4 \end{bmatrix}$ are the corresponding eigenvalue and eigenvector for $\mathbf{A} = \begin{bmatrix} 18 & -2 \\ 12 & 7 \end{bmatrix}$.

Exercise 12.6

Continuing the example above, with

$$\mathbf{A} = \begin{bmatrix} 18 & -2 \\ 12 & 7 \end{bmatrix}$$

find the eigenvector that corresponds to the eigenvalue $\lambda_2 = 15$.

More Eigen-stuff

Exercise 12.7

Determine the eigenvalues and eigenvectors of the following 2x2 matrices.

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

Exercise 12.8

We have two vectors,

$$\mathbf{n} = \begin{bmatrix} -1 \\ 1 \end{bmatrix} \tag{12.1}$$

but

$$\mathbf{z} = \begin{bmatrix} -1 \\ 1.01 \end{bmatrix} \tag{12.2}$$

In other words, the vectors $\mathbf{n}$ and $\mathbf{z}$ point in a very similar direction, but are not perfectly aligned. Now consider a matrix $\mathbf{S}$ given by

$$\mathbf{S} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \tag{12.3}$$

1. On the same axes, plot the vectors $\mathbf{n}$ and $\mathbf{z}$ using MATLAB.
2. Suppose that $\mathbf{n}$ and $\mathbf{z}$ are transformed by $\mathbf{S}$. On the same axes as in the previous part, plot the vectors $\mathbf{Sn}$ and $\mathbf{Sz}$ using MATLAB.
3. Now, we shall see what happens to these vectors under repeated transformations by $\mathbf{S}$. On the same axes as in the previous part, plot the vectors $\mathbf{SSn}$ and $\mathbf{SSz}$ using MATLAB.
4. On the same axes as in the previous part, plot the vectors $\mathbf{SSSn}$ and $\mathbf{SSSz}$ using MATLAB.
5. On the same axes as in the previous part, plot the vectors $\mathbf{SSSSn}$ and $\mathbf{SSSSz}$ using MATLAB.
6. You should find that $\mathbf{n}$ is unaffected by the transformation by $\mathbf{S}$, but $\mathbf{z}$ on the other hand moves farther and farther away. In other words, under repeated transformations by $\mathbf{S}$, $\mathbf{z}$ grew further and further apart from his four friends. Explain what you see in terms of eigenvalues and eigenvectors.

12.2 Properties and Applications of Eigenvalues and Eigenvectors

While most of our work on eigenvalues and eigenvectors has focused on 2D vectors and 2×2 matrices, these ideas extend to higher dimensions as well. The eigenvalues and eigenvectors can be found by solving the characteristic polynomial, or by using the MATLAB `eig` function.

A few words about `eig` are in order. The following command

```
>> [V,D] = eig(A)
```

will return two matrices. The columns of the matrix V are the eigenvectors. D is a diagonal matrix, with the eigenvalues on the diagonal. The first eigenvector is in the first column of V and has a corresponding eigenvalue in the first diagonal entry of D. Each eigenvector is normalized to have a magnitude of 1. The eigenvalues will often "appear" to be sorted according to their size, but this is not necessarily true, and is simply an artifact of the algorithm used to compute them. See the documentation in MATLAB for more details.

Consider an $n \times n$ matrix $\mathbf{A}$. The characteristic polynomial will be a polynomial of degree n in λ, i.e., it will have the form

$$c_n\lambda^n + \cdots + c_1\lambda + c_0 = 0$$

where c_i are constants. This polynomial will have n roots, although some of those roots might be the same (e.g., both roots of the polynomial $\lambda^2 + 2\lambda + 1 = 0$ are -1, so we say $\lambda_1 = -1$ and $\lambda_2 = -1$.) This means that an $n \times n$ matrix has n eigenvalues $\lambda_1, \lambda_2, \cdots, \lambda_n$, where its possible that some eigenvalues are equal.

The following are properties of the eigenvalues (some of these are n-dimensional extensions of what you already saw for 2 dimensions).

- $\text{Tr}(\mathbf{A}) = \lambda_1 + \lambda_2 + \cdots + \lambda_n$

- $\det(\mathbf{A}) = \lambda_1 \lambda_2 \cdots \lambda_n$

- $\mathbf{A}$ is invertible if and only if all eigenvalues are nonzero.
- If the eigenvalues are distinct (none are equal) then the corresponding eigenvectors are linearly independent.
- If a matrix is symmetric, i.e., $\mathbf{A} = \mathbf{A}^T$, then its eigenvalues are real and its eigenvectors are orthogonal to each other.

Exercise 12.9

In this problem, you will get some practice seeing some of the properties above in action. First, create a 3×3 matrix in MATLAB, with any values you'd like and call it A. Alternatively, you can ask MATLAB to generate a 3×3 matrix with random entries using `A = randn(3, 3);`.

1. Use MATLAB's `eig` function to get the eigenvalues and eigenvectors of the matrix.
2. Using MATLAB's `trace` function, confirm that the trace equals the sum of the eigenvalues.
3. Using MATLAB's `det` function, confirm that the determinant equals the product of the eigenvalues, and explain why a square matrix is invertible if and only if all its eigenvalues are nonzero.
4. Generate a new matrix $\mathbf{B} = \mathbf{A}^T\mathbf{A}$ which must be symmetric. Find its eigenvalues and eigenvectors using `eig`, and verify that the eigenvectors are orthogonal.

12.3 Eigenvalues and Eigenvectors in Data Analysis

In the last class you worked on examples involving correlation matrices. Here we will look at covariance matrices, which are related to correlation matrices, except that the entries are not normalized by the standard deviations of the variables. You can think of covariance matrices as measuring the relationship between random quantities, but without normalization. Thus, information about how small or large these data values are will still be preserved in the covariance matrix.

Suppose that we have two different data variables x and y (e.g. corresponding to temperatures in Boston and Sao Paolo), with x_i and y_i being different values in the data set we can define a a matrix $\mathbf{A}$ as follows:

$$\mathbf{A} = \frac{1}{\sqrt{N-1}} \begin{pmatrix} x_1 - \mu_x & y_1 - \mu_y \\ x_2 - \mu_x & y_2 - \mu_y \\ x_3 - \mu_x & y_3 - \mu_y \\ \vdots & \vdots \\ x_N - \mu_x & y_N - \mu_y \end{pmatrix}$$

where μ_x is the mean of the first column, and N is the number of samples (rows). The covariance matrix of x and y is $\mathbf{R} = \mathbf{A}^T\mathbf{A}$. You can think of the entries of this matrix as storing the un-normalized correlations between the temperatures. Because $\mathbf{R}^T = \mathbf{R}$, this matrix is symmetric, and hence has orthogonal eigenvectors.

The eigenvectors and eigenvalues of $\mathbf{R}$ tell us something about how the data are distributed. The eigenvector corresponding to the largest eigenvalue of $\mathbf{R}$, which is also called the *principal eigenvector* of $\mathbf{R}$ points in the direction with the largest variation in the data. The eigenvector corresponding to the second largest eigenvalue points in the direction orthogonal to the principal eigenvector in which there is the second largest amount of variation in the data, and so on (if you have more than 2 dimensional data). The square-root of the eigenvalues tells you about the amount of variation there is in each of those directions. Of course when you only have two different variables in the data set, the matrix $\mathbf{R}$ has only 2 orthogonal eigenvectors.

To illustrate, consider Figure 12.2 which shows the centered (mean subtracted) temperatures of Boston vs Sao Paolo. We have also plotted the two eigenvectors, scaled by the square-root of their corresponding eigenvalues, to illustrate the relative variation of the data along the directions of the two eigenvectors. Notice that the principal eigenvector is in the direction of greatest variation in the data. Figure 13.1 is a similar plot with the temperatures of Boston and Washington DC instead.

The proof of this is optional and will be introduced in the future.

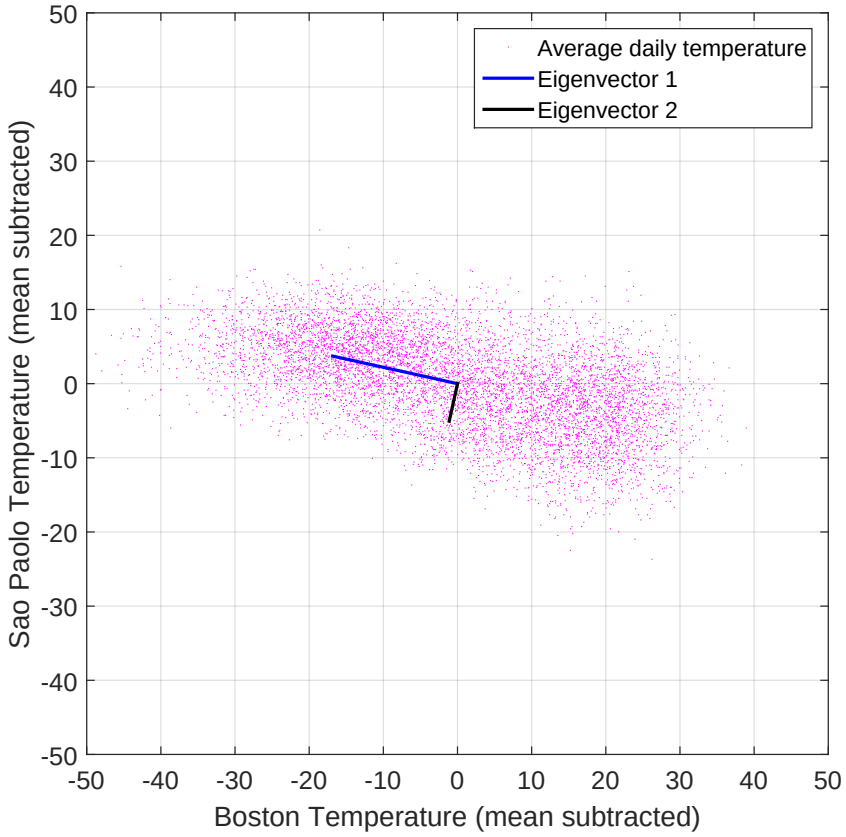


Figure 12.2: Centered average daily temperatures of Boston vs Sao Paolo, with the eigenvectors of the covariance matrix.

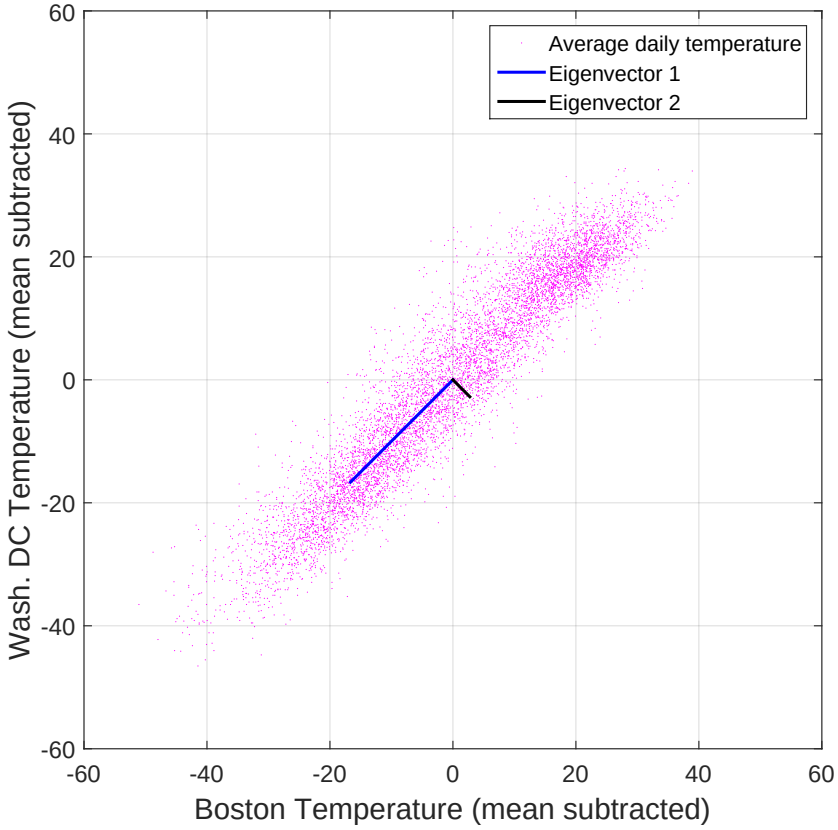


Figure 12.3: Centered average daily temperatures of Boston vs Washington DC, with the eigenvectors of the covariance matrix.

Exercise 12.10

In this next problem, we are going to visualize how the eigenvectors of covariance matrices can tell us about the directions of most variation in 3D data. Load the file `temps_bos_sp_dc.mat` in MATLAB. This file will load 21 years of temperature values for Boston, Sao Paolo and Washington DC. Treat the temperatures of Boston, Sao Paolo, and Washington DC for a given day as a point in a 3D space.

1. Subtract out the mean temperature of each city from the daily temperature data.
2. Make a 3D scatter plot of the data points with the means subtracted out. You will find MATLAB's `plot3` function useful. You may wish to use the `'MarkerSize'` argument

for `plot3` with a marker size of 0.1 or less to make the plots clearer.

3. Construct a covariance matrix for the data and compute its eigenvectors.
4. On the same axes, using `quiver3`, or `plot3`, plot the eigenvectors scaled by the square-root of their corresponding eigenvalues. Use `grid on` to draw grid lines on the axes to improve your visualization.
5. Using the rotate 3D button on the figure window, rotate the image around to see how the eigenvectors tell you about the variation in the data.

12.4 Diagnostic Quiz

Please see Canvas for the quiz questions.

Solution 12.1

1. First we find

$$\mathbf{A} - \lambda\mathbf{I} = \begin{bmatrix} 2-\lambda & 0 \\ 0 & -3-\lambda \end{bmatrix}.$$

And then we compute the determinant

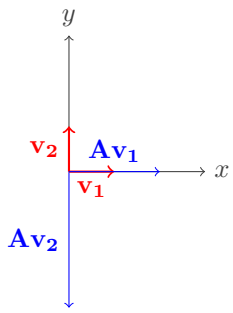
$$\det(\mathbf{A} - \lambda\mathbf{I}) = (2-\lambda)(-3-\lambda).$$

Setting this equal to zero produces the characteristic equation,

$$(2-\lambda)(-3-\lambda) = 0$$

whose roots are, in fact, $\lambda_1 = 2$ and $\lambda_2 = -3$.

2. Here's a plot of $\mathbf{v}_1, \mathbf{v}_2, \mathbf{A}\mathbf{v}_1$ and $\mathbf{A}\mathbf{v}_2$:



When the eigenvalue is negative, the eigenvector is reversed in direction and then scaled.

Solution 12.2

1. Since $\text{tr}(\mathbf{A}) = 2$ and $\det(\mathbf{A}) = -3$, we have

$$\lambda^2 - 2\lambda - 3 = 0$$

so $\lambda_1 = -1$ and $\lambda_2 = 3$.

2. Since $\text{tr}(\mathbf{A}) = 3$ and $\det(\mathbf{A}) = -4$, we have

$$\lambda^2 - 3\lambda - 4 = 0$$

so $\lambda_1 = -1$ and $\lambda_2 = 4$.

Solution 12.3

1.

$$\begin{aligned}\lambda_1 + \lambda_2 &= \frac{tr(\mathbf{A}) + \sqrt{tr(\mathbf{A})^2 - 4det(\mathbf{A})}}{2} + \frac{tr(\mathbf{A}) - \sqrt{tr(\mathbf{A})^2 - 4det(\mathbf{A})}}{2} \\ &= \frac{tr(\mathbf{A})}{2} + \frac{tr(\mathbf{A})}{2} = tr(\mathbf{A})\end{aligned}$$

2.

$$\begin{aligned}\lambda_1\lambda_2 &= \Big(\frac{tr(\mathbf{A})}{2}\Big)^2 + \Big(\frac{tr(\mathbf{A})}{2}\Big)\Big(\sqrt{tr(\mathbf{A})^2 - 4det(\mathbf{A})}\Big) \\ &- \Big(\frac{tr(\mathbf{A})}{2}\Big)\Big(\sqrt{tr(\mathbf{A})^2 - 4det(\mathbf{A})}\Big) - \Big(\frac{\sqrt{tr(\mathbf{A})^2 - 4det(\mathbf{A})}}{2}\Big)^2 \\ &= \frac{tr(\mathbf{A})^2}{4} - \frac{tr(\mathbf{A})^2 - 4det(\mathbf{A})}{4} \\ &= det(\mathbf{A})\end{aligned}$$

3. Symmetric means $b = c$ and real λ means:

$$tr(\mathbf{A})^2 - 4det(\mathbf{A}) \geq 0$$

$$(a+d)^2 - 4(ad - bc) \geq 0$$

$(a+d)^2 - 4(ad-bc) = a^2 + d^2 + 2ad - 4ad + 4bc = a^2 + d^2 - 2ad + 4b^2 = (a-d)^2 + 4b^2$

Squares of real numbers are positive, so $tr(\mathbf{A})^2 - 4det(\mathbf{A}) \geq 0$ and the eigenvalues λ are real.

Solution 12.4

Using the fact that $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$, we see that

$$\mathbf{A}(c\mathbf{v}) = c\mathbf{A}\mathbf{v} = c\lambda\mathbf{v} = \lambda(c\mathbf{v})$$

and therefore $c\mathbf{v}$ is also an eigenvector.

Solution 12.5

First we compute the left-hand side

$$\mathbf{A}\mathbf{v}_1 = \begin{bmatrix}18 & -2 \\ 12 & 7\end{bmatrix}\begin{bmatrix}1 \\ 4\end{bmatrix} = \begin{bmatrix}10 \\ 40\end{bmatrix}$$

and the right-hand side

$$\lambda_1\mathbf{v}_1 = 10\begin{bmatrix}1 \\ 4\end{bmatrix} = \begin{bmatrix}10 \\ 40\end{bmatrix}.$$

Fortunately, they are equal.

Solution 12.6

First we compute

$$\mathbf{A} - \lambda_2 \mathbf{I} = \begin{bmatrix} 18-15 & -2 \\ 12 & 7-15 \end{bmatrix} = \begin{bmatrix} 3 & -2 \\ 12 & -8 \end{bmatrix}.$$

Now, letting $\mathbf{v}_2 = \begin{bmatrix} a \\ b \end{bmatrix}$, we are trying to solve

$$\begin{bmatrix} 3 & -2 \\ 12 & -8 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

which produces the equations

$$3a - 2b = 0 \text{ and } 12a - 8b = 0.$$

(These equations give the same information since the rows of $(\mathbf{A} - \lambda_2 \mathbf{I})$ are linearly dependent.) This gives $b = \frac{3}{2}a$, so $\mathbf{v}_2 = \begin{bmatrix} a \\ \frac{3}{2}a \end{bmatrix}$ for any value of a. Picking $a = 2$, we have $\mathbf{v}_2 = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$.

Solution 12.7

1. $$\lambda = 5, 2$$

   and

   $$\mathbf{v} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

2. $$\lambda = -1, 4$$

   and

   $$\mathbf{v} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

Solution 12.8

See Figure 12.1. $\mathbf{n}$ is an eigenvector of $\mathbf{S}$ with an eigenvalue of 1, so it is unchanged by the transformation $\mathbf{S}$. However, $\mathbf{z}$ is not an eigenvector of $\mathbf{S}$, so it changes each time the transformation $\mathbf{S}$ is applied, and the change accelerates as it diverges from the eigenvector $\mathbf{n}$.

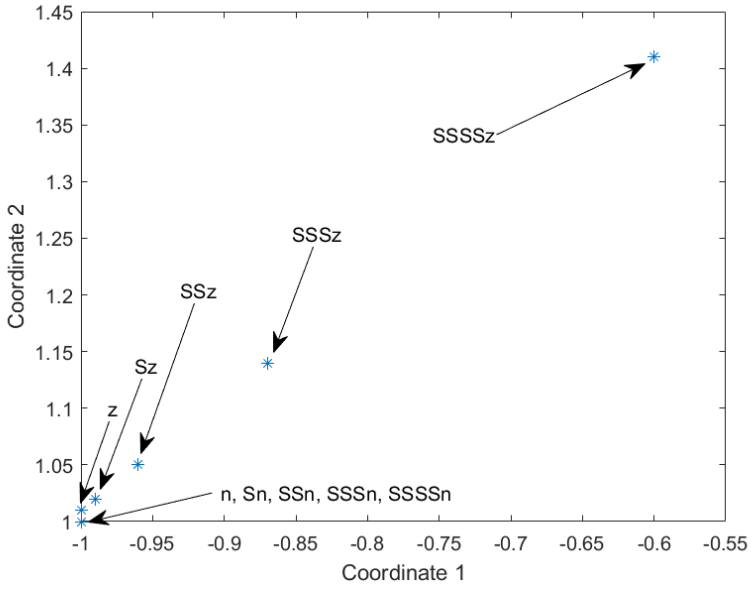


Figure 12.1: Plot for Exercise 9.

Solution 12.9

1. `A=randn(3,3);[V,D]=eig(A)`
2. `trace(A)-sum(diag(D))`
3. `det(A)-prod(diag(D))`

   The product of all eigenvalues equals the determinant of the matrix. If any eigenvalue is zero, the determinant is zero and the matrix is non-invertible; if all eigenvalues are nonzero, the determinant is nonzero and the matrix is invertible.

4. `B=A'*A;[V D]=eig(B);V'*V` gives the identity matrix, showing that the eigenvectors are orthogonal (and of unit length).

Solution 12.10

1. `bn=b-mean(b);sn=s-mean(s);wn=w-mean(w);`
2. ```
   plot3(bn,sn,wn,'.','MarkerSize',0.1)
   xlabel('Boston temperature (mean subtracted)')
   ylabel('Sao Paolo temperature (mean subtracted)')
   zlabel('Wash. D.C. temperature (mean subtracted)'
   ```
   ```

3.
```
A=1/sqrt(length(b)-1)*[bn,sn,wn];
R=A'*A;
[V,D]=eig(R)
```

4.
```
plot3(bn,sn,wn,'.','MarkerSize',0.1)
Vs=V.*sqrt(diag(D))
hold on
plot3([0,Vs(1,1)],[0 Vs(2,1)],[0 Vs(3,1)],'LineWidth',2)
plot3([0,Vs(1,2)],[0 Vs(2,2)],[0 Vs(3,2)],'LineWidth',2)
plot3([0,Vs(1,3)],[0 Vs(2,3)],[0 Vs(3,3)],'LineWidth',2)
grid on
axis equal
```

   See Figure 14.1, which has the first eigenvector clearly aligned with the direction of greatest variation.

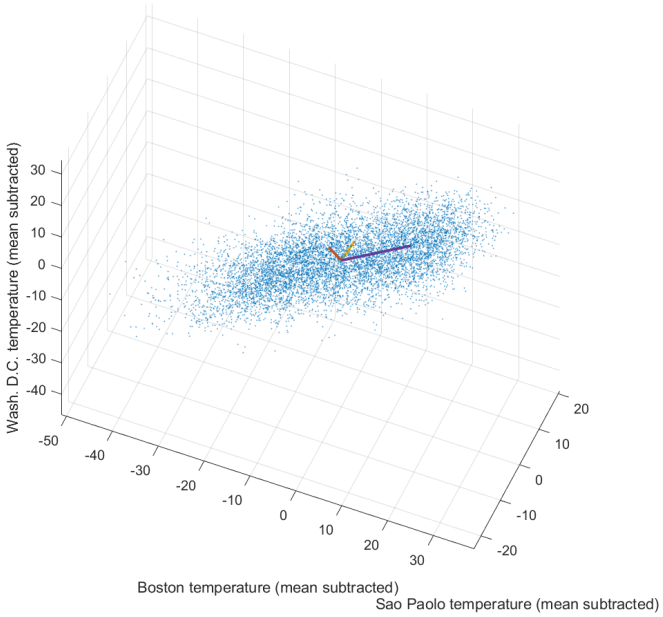


Figure 12.4: Temperatures and eigenvectors.

Chapter 13

Day 7: EVD and PCA

13.1 Schedule

- 0900-0930: Debrief
- 0930-1015: Eigenvalue Decomposition (EVD)
- 1015-1030: Coffee
- 1030-1115: PCA and Maximum Variance
- 1115-1210: Conceptual PCA and PCA blog post
- 1210-1225: Review and Preview
- 1225-1230: Survey

13.2 Debrief [15 mins]

In the last class and in the take-home exercise, you worked on a number of different exercises involving eigenvalues and eigenvectors.

Exercise 13.1

1. With your table, identify a list of key concepts/take home messages/things you learned in the last class and take-home assignment.
2. Try to resolve your confusions with the folks at your table and by talking to an instructor.

13.3 Eigenvalue Decomposition (EVD) [45 mins]

The eigenvalue decomposition, also known as the eigendecomposition, is an operation on matrices in which a square matrix is expressed as a product of matrices made up of its eigenvalues and eigenvectors. It can be used to find inverses and powers of matrices, as well as to derive some important results in data analysis. For instance, in a prior exercise, you saw that the eigenvector corresponding to the largest eigenvalue of a covariance matrix was in the direction of greatest variance in your data set. This property can be proved using the eigendecomposition.

The eigenvalue decomposition is also helpful in dimensionality reduction, which is a process where we can represent higher-dimensional vectors as a linear combination of a smaller number of vectors than dimensions – an example of which you saw in a previous exercise where you represented pictures of peoples faces using a linear combination of vectors. The eigendecomposition is also often used to change coordinate systems.

The Big Idea

Assume that a square $n \times n$ matrix $\mathbf{A}$ has n linearly independent eigenvectors $\mathbf{v}_i$ with corresponding eigenvalues λ_i, i.e.

$$\mathbf{A}\mathbf{v}_i = \lambda_i \mathbf{v}_i \; i = 1, 2, \ldots, n$$

Instead of thinking of these eigenvalues and eigenvector separately, let's package them into matrices as follows:

$$[\mathbf{A}\mathbf{v}_1 \; \mathbf{A}\mathbf{v}_2 \; \ldots \; \mathbf{A}\mathbf{v}_n] = [\lambda_1\mathbf{v}_1 \; \lambda_2\mathbf{v}_2 \; \ldots \; \lambda_n\mathbf{v}_n]$$

Properties of matrix multiplication suggests that we can re-write this matrix equation in the form

$$\mathbf{A}[\mathbf{v}_1 \; \mathbf{v}_2 \; \ldots \; \mathbf{v}_n] = [\mathbf{v}_1 \; \mathbf{v}_2 \; \ldots \; \mathbf{v}_n] \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \ldots & 0 \\ 0 & 0 & \lambda_n \end{bmatrix}$$

where the last matrix has each eigenvalue on the diagonal. If we now define

$$\begin{aligned} \mathbf{V} &= [\mathbf{v}_1 \; \mathbf{v}_2 \; \ldots \; \mathbf{v}_n] \\ \mathbf{D} &= \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \ldots & 0 \\ 0 & 0 & \lambda_n \end{bmatrix} \end{aligned}$$

then the previous equation becomes

$$\mathbf{AV} = \mathbf{VD}$$

Since we assumed that the eigenvectors are linearly independent this implies that the columns of $\mathbf{V}$ are linearly independent which in turn implies that the inverse of $\mathbf{V}$ exists. We can there fore write

$$\mathbf{A} = \mathbf{V}\mathbf{D}\mathbf{V}^{-1} \tag{13.1}$$

where the matrix $\mathbf{V}$ has the i-th eigenvector of $\mathbf{A}$ as its i-th column, and $\mathbf{D}$ is a diagonal matrix with the i-th eigenvalue of $\mathbf{A}$ as its ii-th entry. This expression is known as the *eigendecomposition* of $\mathbf{A}$. In the special case where $\mathbf{A}$ is symmetric, the eigenvalues are real, and the eigenvectors are mutually orthogonal so that

$$\mathbf{V}^{-1} = \mathbf{V}^T,$$

which is a property of $n \times n$ matrices whose column vectors are mutually orthogonal and have a length of 1 (i.e., the column vectors are orthonormal).

Exercise 13.2

1. Consider the following 2×2 matrix $\mathbf{A}$.

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

   By hand, compute its eigenvectors and eigenvalues, determine the matrices $\mathbf{V}$, $\mathbf{D}$, and $\mathbf{V}^{-1}$,

and confirm that (13.1) is correct. Use MATLAB to confirm your results by computing » [V,D]=eig(A). **Note**: you should normalize each of your eigenvectors to be unit length.

Exercise 13.3

1. The eigendecomposition can be used to change basis as follows. Consider the matrix $\mathbf{A}$ from the previous exercise as a transformation matrix.
   a) How does the matrix $\mathbf{A}$ transform the vector $\mathbf{w} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$? Draw both $\mathbf{w}$ and $\mathbf{Aw}$ on an xy-coordinate plane.
   b) Draw both eigenvectors of $\mathbf{A}$ on this coordinate plane.
   c) Decompose the vector $\mathbf{w}$ as a linear combination of both eigenvectors. You should be able to do this with a matrix-vector multiply. You are expressing the vector in a new basis.
   d) Scale each component by the relevant eigenvalue.
   e) Undo the decomposition to return to the original basis.
   f) What just happened?

Exercise 13.4

One thing that the eigendecomposition helps us compute is how to raise $\mathbf{A}$ to an integer power, without going through the process of repeated multiplication.

1. Using eigendecomposition, show the following is true
$$\mathbf{A}^2 = \mathbf{V}\mathbf{D}^2\mathbf{V}^{-1} \tag{13.2}$$
and confirm this result using the matrix from earlier the earlier exercise. Note that for any diagonal matrix $\mathbf{D}$, $\mathbf{D}^k$ is another diagonal matrix whose ii-th entry equals the ii-th entry of $\mathbf{D}$ raised to the k-th power. Hence computing $\mathbf{D}^n$ is not computationally difficult - you just raise each diagonal entry to the n-th power.
2. Show that the following is also true
$$\mathbf{A}^n = \mathbf{V}\mathbf{D}^n\mathbf{V}^{-1}$$

13.4 Principal Components Analysis (PCA)

In the night assignment you explored, in a graphical manner, the relationship between the eigenvectors of the covariance matrix and the distribution of the data. For instance, you looked at the daily temperature values in Boston versus Sao Paolo and the daily temperatures in Boston versus Washington D.C.

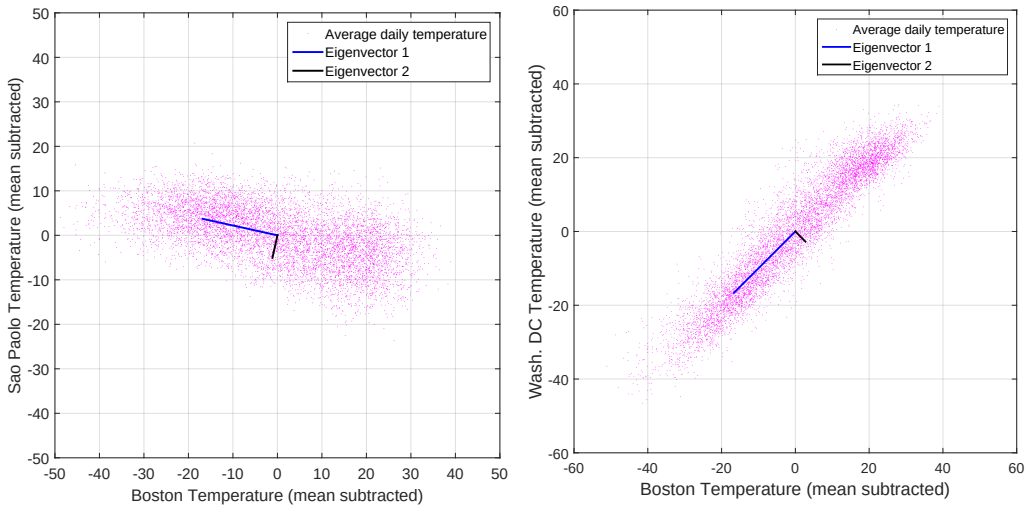


Figure 13.1: Centered average daily temperatures of Boston vs Sao Paolo (left) and Boston vs Washington DC, with the eigenvectors of the covariance matrix.

From visually inspecting these figures we saw that eigenvector 1, which corresponded to the larger of the two eigenvalues, seemed to be pointing in the direction where the data exhibited the most variability (i.e., the data was most spread out along this direction). You also looked at this for a 3D dataset consisting of the temperatures from Boston, Sao Paolo, and Washington DC.

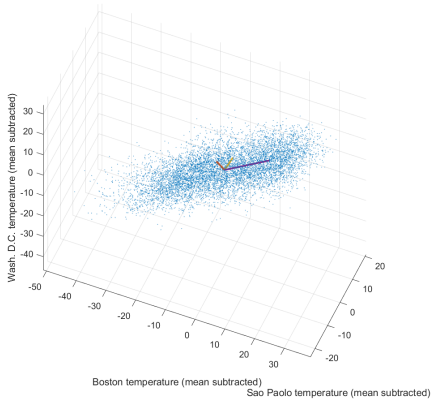


Figure 13.2: Temperatures and eigenvectors for Boston, Sao Paolo, and Washington DC

In this 3D dataset, we see the same phenomenon: that the principal eigenvector points along the direction of maximum variation in the data. It turns out that this phenomenon will hold no matter the dimensionality of the data (it works for 4D datasets, 10D datasets, and even datasets with 1,000s of dimensions)! This fact provides the basis for the principal components algorithm. In PCA, instead of working with the data in its original form, we express it in a basis given by eigenvectors of the covariance matrix that have the largest eigenvalues. We can understand the properties of using this basis through two key properties.

- *Property 1*: the principal eigenvectors of the covariance matrix will maximize the variance of the data when the data is projected onto these vectors (we can think of vectors that capture large variation in the data as representing important properties of the data).
- *Property 2*: the principal eigenvectors of the covariance matrix will allow us, in a particular sense, to optimally compress our data. That is, we will be able to recover the original data with the highest possible accuracy from the projections of the data onto the principal eigenvectors.

The power of PCA lies in its ability to achieve both of the properties described above simultaneously. For this reason, the principal components of a dataset will act as keys to unlocking the secrets lurking in the data! **Today we will be exploring property 1, and in the night assignment you will also be exploring property 2**.

The Principal Eigenvector as the Direction of Maximum Variance

The graphs of the daily temperature data show, graphically, that the principal eigenvector of the covariance matrix corresponds to the direction of maximum variation in the data. In this section we'll be formalizing this result. We've decided to structure this part of the day assignment as an extended exercise where you will be working through the proof of this fact step-by-step. While there are many ways to do this proof, we'll be walking you through one way that will connect well with the ideas we've been exploring in the last week or so of the course. We recommend that you do a part of the proof, check it against the solutions and then move onto the next piece.

Before getting started, let's look at some material from night 6 that shows that the covariance matrix can be computed using matrix multiplication.

Suppose that we have two different data variables x and y (e.g. corresponding to temperatures in Boston and Sao Paolo), with x_i and y_i being different values in the data set we can define a a matrix $\mathbf{A}$ as follows:

$$\mathbf{A} = \frac{1}{\sqrt{N-1}} \begin{pmatrix} x_1 - \mu_x & y_1 - \mu_y \\ x_2 - \mu_x & y_2 - \mu_y \\ x_3 - \mu_x & y_3 - \mu_y \\ \vdots & \vdots \\ x_N - \mu_x & y_N - \mu_y \end{pmatrix} \tag{13.3}$$

where μ_x is the mean of the first column, and N is the number of samples (rows). The covariance matrix of x and y is $\mathbf{R} = \mathbf{A}^T\mathbf{A}$. You can think of the entries of this matrix as storing the un-normalized correlations between the temperatures. Because $\mathbf{R}^T = \mathbf{R}$, this matrix is symmetric, and hence has orthogonal eigenvectors.

Let's assume that we are given a dataset with n samples and d dimensions (instead of just 2 dimensions as shown above). We can transform it into the form given in Equation 13.3 by subtracting the mean from each column and dividing the entire matrix by $\sqrt{N-1}$. We now have a mean-centered data matrix $\mathbf{A}$ with n rows and d columns and the covariance matrix of our data is given by $\mathbf{A}^\top\mathbf{A}$.

Exercise 13.5

Our overall goal is to show that if we take a unit vector $\mathbf{u}$, project our mean-centered data onto it (as $\mathbf{Au}$), and examine the variance of the projected data, that this variance is largest when $\mathbf{u}$ is the principal eigenvector of the covariance matrix $\mathbf{A}^\top\mathbf{A}$.

1. First we'll write down an expression for the variance of $\mathbf{Au}$ (we'll write this as $Var[\mathbf{Au}]$) as a matrix multiplication. We'll do this step together (i.e., we'll show you how to do it). For this part of the exercise you should make sure you understand the steps we performed.

   If $\mathbf{A}$ is in the form given in Equation 13.3, then $\mathbf{Au}$ will have 0 mean (since $\mathbf{Au}$ is a linear combination of columns with 0 mean). Using the same logic that led us to conclude that $\mathbf{A}^\top\mathbf{A}$ is the covariance matrix of the data, $(\mathbf{Au})^\top(\mathbf{Au})$ will give us the variance of the data projected onto $\mathbf{u}$ (remember that variance is just a special case of covariance where we are comparing a quantity to itself). It's worth noting that since $\mathbf{Au}$ is a vector, the expression $(\mathbf{Au})^\top(\mathbf{Au})$ is known as the inner product, which is really the same as the dot product (that is, $(\mathbf{Au})^\top(\mathbf{Au}) = \mathbf{Au} \cdot \mathbf{Au}$). Thus, the variance is given by the following equation.

$$\begin{aligned} Var[\mathbf{Au}] &= (\mathbf{Au})^\top(\mathbf{Au}) \\ &= \mathbf{u}^\top\mathbf{A}^\top\mathbf{Au} \quad \text{note: we are applying the rule that } (\mathbf{AB})^\top = \mathbf{B}^\top\mathbf{A}^\top \end{aligned}$$

2. Substitute the eigenvalue decomposition, $\mathbf{VDV}^\top$, for the covariance matrix $\mathbf{A}^\top\mathbf{A}$ (since $\mathbf{A}^\top\mathbf{A}$ is symmetric and real, we can substitute $\mathbf{V}^\top$ for the inverse of $\mathbf{V}$ in the eigenvalue decomposition).

3. Define the vector $\mathbf{y} = \mathbf{V}^\top\mathbf{u}$ and substitute it into the expression from part 2.

4. Expand out the expression in part 3 so that it is in terms of the squares of the elements of $\mathbf{y}$ and the diagonal entries of $\mathbf{D}$ in order of largest to smallest.
5. Show that $\mathbf{y}$ is a unit vector by taking the inner product with itself and showing that it is equal to 1 (recall that the inner product is the same as the dot product). *Hint:* $\mathbf{V}\mathbf{V}^\top = \mathbf{I}$ *since* $\mathbf{V}$ *is orthonormal and has* d *linearly independent columns.*
6. Argue that since $\mathbf{y}$ is a unit vector (which implies $\sum_{i=1}^{d} y_i^2 = 1$), that the expression in part 4 is maximized when $y_i = 1$ when i is the index of the principal eigenvector and $y_i = 0$ when i is any other index. To get a feel for why this is true, try writing out a specific case where, perhaps, $\mathbf{y}$ has two or three dimensions.
7. Show that we achieve the value of $\mathbf{y}$ in part 5 (that is where $y_i = 1$ when i is the index of the principal eigenvector and $y_i = 0$ when i is any other index) when $\mathbf{u}$ is the principal eigenvector of $\mathbf{A}^\top\mathbf{A}$.
8. What have you just shown?!? Make sure you have a sense of what you just did (don't get lost in the mathematical symbols).

Beyond the first principal component

We've now gone into depth in understanding the first principal component and its amazing property of maximizing variance. The second principal component is simply going to be the direction that maximizes variance subject to the requirement that it is orthogonal to the first principal component. With a slight modification to your proof you can show that the second principal component will be in the direction of the eigenvector with the second largest eigenvalue. The trend continues for other principal components (i.e., the ith principal component is the eigenvector with the ith largest eigenvalue).

Applications of PCA (thinking it through conceptually)

In this section you're going to be thinking about what the PCA algorithm might do when applied in different domains. The focus of this section will be on trying to understand at a conceptual level what might happen when we apply PCA. In the next section, you'll be reading through an example of applying PCA to some actual data.

Exercise 13.6

For each application, hypothesize what the first principal component might be. That is, for each particular scenario what would the direction be that maximizes the variance of the data projected onto that direction? What might the second principal component be (that is a vector orthogonal to the first that maximizes the variance of the data)?

1. Consider a dataset consisting of ratings from n users of m movies. Let's assume that the ratings are numerical and are on a scale of 1 to 5 (5 being the best). Consider some collection of movies (they could be some specific movies or you could just think of movie genres) and a particular population of users (could be college students, QEA professors, or just the general population). Draw the data matrix $\mathbf{A}$ and label the rows and columns (e.g., with movies or users). In a qualitative sense, make a prediction as to what the first principal component would look like for this dataset. What might the second principal component look like? No numbers... just guess at which dimensions would be positive, negative, or close to 0 for your principal components.

2. Consider a dataset consisting of the prevalence of the flu in various parts of the US. The CDC maintains an animated map of the flu activity over time, which you can (and should) access at https://www.cdc.gov/flu/weekly/usmap.htm. To simplify this data, let's think about the number of flu cases in each of the six major major geographical regions of the US.

If we think about our data matrix as consisting of a row for each week of measured flu activity and each column as a region of the US, in a qualitative sense, make a prediction as to what the first principal component would look like for this dataset. What might the second principal component look like? No numbers... just guess at which dimensions would be positive, negative, or close to 0 for your principal components.

Exercise 13.7

With your table-mates, read through this post that shows the application of PCA to understanding the US political leanings (if you are viewing this in DropBox preview and can't click the link, go to http://bit.ly/37n9qwe). Before, starting here are some process suggestions.

- Checkin with folks at your table as to how they'd like to go through this document (e.g., read

the entire thing individually and come together and ask questions, read it individually but stop after each major section to ask questions, read it aloud as a table).

- If you don't understand something, you can either call over an instructor or note your confusion on the whiteboard and keep going (e.g., if its something that doesn't impede your understanding of the main points in the article).

Solution 13.2

1. The eigenvalues are $\lambda_1 = 1$ and $\lambda_2 = 3$ with corresponding eigenvectors $\mathbf{v}_1 = \frac{1}{\sqrt{2}}\begin{bmatrix}-1\\1\end{bmatrix}$ and $\mathbf{v}_2 = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\1\end{bmatrix}$. This gives

$$\mathbf{V} = \frac{1}{\sqrt{2}}\begin{bmatrix}-1 & 1\\1 & 1\end{bmatrix},\ \mathbf{D} = \begin{bmatrix}1 & 0\\0 & 3\end{bmatrix},\ \mathbf{V}^{-1} = \sqrt{2}\begin{bmatrix}-1/2 & 1/2\\1/2 & 1/2\end{bmatrix}.$$

   and if you multiply them all together you will get the original matrix $\mathbf{A}$. Running "eig" in MATLAB gives the same eigenvalues and eigenvectors, although every eigenvector could be multiplied by -1. MATLAB may also place your eigenvalues and eigenvectors in a different order.

Solution 13.3

1. The vector becomes $\begin{bmatrix}5\\4\end{bmatrix}$.
2. The eigenvectors were $\mathbf{v}_1 = \frac{1}{\sqrt{2}}\begin{bmatrix}-1\\1\end{bmatrix}$ and $\mathbf{v}_2 = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\1\end{bmatrix}$.
3. Decomposing the vector $\mathbf{w}$ as linear combination of the eigenvectors is equivalent to solving

$$\mathbf{V}\mathbf{c} = \mathbf{w}$$

   for the vector $\mathbf{c}$. This is the coordinates of the vector $\mathbf{w}$ in the new basis. You should find that $\mathbf{c} = \sqrt{2}\begin{bmatrix}-0.5\\1.5\end{bmatrix}$.
4. We multiply the first component by 1 and the second component by 3 to give $\sqrt{2}\begin{bmatrix}-0.5\\4.5\end{bmatrix}$.
5. In order to undo the change of basis we hit this vector with $\mathbf{V}$ which gives $\begin{bmatrix}5\\4\end{bmatrix}$ as expected.
6. The eigendecomposition can be be thought of as a change of basis followed by a scaling matrix followed by the change back to the original basis.

Solution 13.4

1. Since $\mathbf{A} = \mathbf{V}\mathbf{D}\mathbf{V}^{-1}$, we know that

$$\mathbf{A}^2 = \mathbf{V}\mathbf{D}\mathbf{V}^{-1}\mathbf{V}\mathbf{D}\mathbf{V}^{-1} = \mathbf{V}\mathbf{D}^2\mathbf{V}^{-1}.$$

2. Similar reasoning to the previous problem shows that

$$\mathbf{A}^n = \mathbf{V}\mathbf{D}^n\mathbf{V}^{-1}$$

Solution 13.5

1. Solution is already given in the problem

2.

$$Var[\mathbf{Au}] = \mathbf{u}^\top \mathbf{VDV}^\top \mathbf{u}$$

3.

$$\begin{aligned} Var[\mathbf{Au}] &= (\mathbf{V}^\top \mathbf{u})^\top \mathbf{D}(\mathbf{V}^\top \mathbf{u}) \\ &= \mathbf{y}^\top \mathbf{Dy} \end{aligned}$$

4.

$$\begin{aligned} Var[\mathbf{Au}] &= \mathbf{y}^\top \mathbf{Dy} \\ &= \mathbf{y}^\top \begin{bmatrix} y_1 D_{1,1} \\ y_2 D_{2,2} \\ \vdots \\ y_d D_{d,d} \end{bmatrix} \\ &= \sum_{i=1}^{d} y_i^2 D_{i,i} \end{aligned}$$

5.

$$\begin{aligned} \mathbf{y}^\top \mathbf{y} &= (\mathbf{V}^\top \mathbf{u})^\top (\mathbf{V}^\top \mathbf{u}) \\ &= \mathbf{u}^\top \mathbf{VV}^\top \mathbf{u} \\ &= \mathbf{u}^\top \mathbf{u} \\ &= 1 \end{aligned}$$

6. If we choose $y_i = 1$ where i is the index of the principal eigenvector, then the expression in part 4 will give us $D_{i,i}$. Any other choice of $\mathbf{y}$ will result in some weighted combination of the eigenvalues (the diagonal elements of $\mathbf{D}$) where the weights are all positive and add up to 1. It is easy to see that putting any weight on a non-maximal eigenvalue will result in a lower variance as computed by the expression in part 4.

7. Since $\mathbf{y} = \mathbf{V}^\top \mathbf{u}$, y_i is the dot product of $\mathbf{u}$ and the ith eigenvector, $\mathbf{v_i}$, with $\mathbf{u}$. Since we assume all of the eigenvectors are unit vectors and mutually orthogonal, if we set $\mathbf{u}$ to be the principal eigenvector of $\mathbf{A}^\top \mathbf{A}$, then the dot product of $\mathbf{u}$ and $\mathbf{v_i}$ will be 1 for i corresponding to the principal eigenvector and 0 for all other indices.

8. You just showed that the direction along the principal eigenvector of the covariance matrix maximizes the variance of the projected data. That's pretty cool!

Chapter 14

Night 7: PCA and Eigenfaces

Learning Objectives

Concepts

- Understand the connection between PCA and eigenvalues and eigenvectors.
- Understand how to use PCA to carry out data compression.
- Understand the idea of using Eigenfaces to do facial recognition.

MATLAB skills

- Use "eig" to carry out a PCA.
- Implement facial recognition using PCA.
- Determine the accuracy of PCA for different numbers of principal components.

14.1 Principal Component Analysis Revisited

As we in class, PCA is an algorithm in which we express our original data along the eigenvectors corresponding to the largest eigenvalues of the covariance matrix. We examined the property of PCA that it if we project our data onto these vectors, this will lead to maximizing the variance of the projected data. To refresh your memory further, here is the temperature plot for Boston, Sao Paolo, and Washington DC.

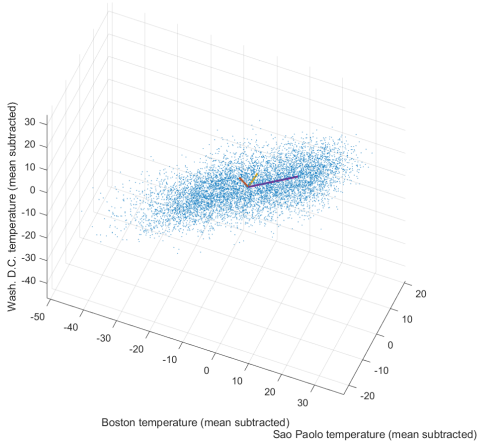


Figure 14.1: Temperatures in three cities and the eigenvectors of the covariance matrix.

We also briefly mentioned a second property of PCA, which is that it can be thought of as an optimal way to compress our data down to a smaller set of numbers. This idea, also known as dimensionality reduction, is going to be a view that we explore in this assignment. If you'd like, here are a few external resources on PCA:

- http://www.cs.otago.ac.nz/
- http://dai.fmph.uniba.sk/courses/ml/sl/PCA.pdf
- https://deeplearning4j.org/eigenvector/linear
- http://www.cerebralmastication.com/2010/09/principal-component-analysis-pca-vs-ordinary-least-squares-ols-a-visual-explination/
- http://stats.stackexchange.com/questions/2691/making-sense-of-principal-component-analysis-eigenvectors-eigenvalues/

PCA in two dimensions

In general, PCA is conducted on data that is mean-centered (i.e., the data has had the mean of each variable subtracted out). To refresh your memory of PCA and scaffold the introduction of the view of PCA as compressing a dataset, let's think about a simple example data set $\mathbf{D}$.

$$\mathbf{D} = \begin{bmatrix} -1 & 3 \\ 1 & 4 \\ 3 & 4 \\ 7 & 5 \\ 10 & 9 \end{bmatrix} \tag{14.1}$$

Exercise 14.1

1. Create a plot of $\mathbf{D}$ as a set of points in the xy-plane.
2. Define a matrix $\tilde{\mathbf{D}}$ which is the mean-centered version of $\mathbf{D}$ and plot $\tilde{\mathbf{D}}$ as a set of points in the xy-plane
3. The principal components ($\mathbf{p_1}$ and $\mathbf{p_2}$) are the eigenvectors of the covariance matrix of the mean-centered $\tilde{\mathbf{D}}$. Compute $\mathbf{p}_1$ and $\mathbf{p}_2$ and plot them on top of the mean-centered data.
4. Compute the projection of your data onto the eigenvector which corresponds to the largest eigenvector, which in this case is $\mathbf{p}_2$. This is the "reduced dimenstionality" version of your data, called $\mathbf{B}$, which only include information about the projection along $\mathbf{p}_2$. (We reduced the 2-dimensional data to 1-dimensional data.) Plot the original data $\mathbf{D}$ and the reduced data $\mathbf{B}$.

Exercise 14.2

1. Can you recreate $\mathbf{D}$ perfectly from $\mathbf{B}$?
2. What would have happened if you had created $\mathbf{B}$ using only information about the values along $\mathbf{p_1}$ instead of $\mathbf{p_2}$?
3. How might you quantify how well you can represent $\mathbf{D}$ in this reduced dimensionality form?
4. If you received a new piece of data, how would you go about representing this as a linear combination of $\mathbf{p_1}$ and $\mathbf{p_2}$?

Data Compression via PCA

In this exercise you will perform a simple data compression exercise, similar to the one you did in a previous Night assignment. You will use temperature data from 3 cities over 10 years, as training data and use it to compress a year's worth of temperature data from 3 cities into a 2×365 matrix. In other words, you will represent 3×365 numbers (daily temperature data from 3 cities over 1 year), using 2×365 values. This compression is lossy, in that you will loose some information. However, by representing the data along the two most significant eigenvectors of the covariance matrix, you can reduce this data loss, because these two directions capture the bulk of the variation in the data set. Please note that while we have laid out the steps you need to take here quite explicitly, it is important for you to fully understand what each step does. You will be using very similar steps in your project.

Exercises

Exercise 14.3

1. Load the file `avg_temperatures_pt2.mat`. You will have 6 data vectors in your workspace. `b_tr, w_tr, s_tr` which represent 10 years of training data for the average daily temperatures in Boston, Washington DC, and Sao Paolo, respectively. The vectors `b_new, w_new, s_new` represent an additional year of data for the three cities – this is the data that you will compress using statistical knowledge of the previous 10 years of data. Create a covariance matrix $\mathbf{R}$ using the 10 years worth of temperature data from Boston, Washington DC and Sao Paolo (in that order).
2. Perform an eigendecomposition of the matrix $\mathbf{R}$, and make a new matrix $\mathbf{V}_p$ which has the 2 eigenvectors corresponding to the 2 largest eigenvalues of $\mathbf{R}$. You should use MATLAB's

`eig` function. Let these eigenvectors be $\mathbf{v}_1$ and $\mathbf{v}_2$.

3. Create centered (i.e. subtract the mean), versions of the new temperature data vectors, and create a 3×365 matrix $\mathbf{T}$ which has the centered temperatures of Boston, Washington DC and Sao Paolo as its rows (in that order). This matrix is a representation of the data you are now going to compress. Let the i-th column of $\mathbf{T}$ be $\mathbf{t}_i$.

4. Take the dot product of each column of the matrix $\mathbf{T}$ (which is a vector of the temperature of Boston, Washington DC and Sao Paolo for a given day) with the two eigenvectors in matrix $\mathbf{V}_p$, and save the values. Let this quantities be called α_{1i} and α_{2i}. In other words,

$$\alpha_{1i} = \mathbf{v}_1^T \mathbf{t}_i$$
$$\alpha_{2i} = \mathbf{v}_2^T \mathbf{t}_i$$

   You can do this using matrix multiplications.

   You should now have 365 different values for α_{1i} and α_{2i}, which are a compressed representation of 3×365 different temperature values. Moreover, these values are the components of the temperature data that lie in the directions of the two eigenvectors of the covariance matrix corresponding to the largest eigenvalues. From what we saw in the previous two classes, these vectors represent the two orthogonal directions in the data that have the most amount of variation, and hence the most "important" directions. Of course, there is a third direction (since the temperature vectors live in a 3-dimensional space), which we are discarding. But since this is the direction in which there is the least amount of variation in the data set, we do not lose too much information.

5. You can now check how well your compression worked, by using the values of α_{1i} and α_{2i} to reconstruct 365 different 3×1 vectors each representing the temperatures for the three cities over the 365 days. Let $\hat{\mathbf{t}}_i$ represent the reconstructed temperature vector on the i-th day. Using what you know about projections onto orthonormal vectors, reconstruct $\mathbf{t}_i$ using α_{1i}, α_{2i}, $\mathbf{v}_1$ and $\mathbf{v}_2$. Repeat this for all 365 days.

6. On the same axes, plot the original and reconstructed temperature for Boston. Repeat this for Washington DC and Sao Paolo. Observe how close the reconstructions are, for the different data sets.

7. How accurately do you think you can represent the data if you used 3 eigenvectors instead of 2?

8. If you feel inspired, repeat the above with temperature data for four different cities, and 2 or 3 different eigenvectors.

While this example can be thought of as a “toy” example where we are representing 3 dimensional data using 2 dimensions, there are many applications for which there may be many more dimensions in the data

for which accurate representations can be made using only a few dimensions. Additionally, you should note that such dimensionality reduction techniques are not just useful in compression, but they are also useful in speeding up computation. We can often get away with analyzing data over a small number of important dimensions, and this is an important technique when we deal with large amounts of data. Overall, these class of techniques is called Principal Component Analysis (PCA), since we are performing analysis along a few principal component directions of the data.

14.2 Face Data Compression via PCA

You are now ready to start applying PCA to face data. You have already seen this in a previous class assignment, except in that assignment you had the help of a genie. Load the MATLAB files `classdata_train.mat` and `classdata_test.mat`. These are the training and test datasets with photos of your classmates. The file contains some images and as well as the identity of the person in each image (coded as an integer from 1 to 89).

Remember that the principal eigenvectors of the covariance matrix tell you the directions of greatest variation in a data set and also the directions that optimally compress our data. Your job is to use the training images to build a model for your faces such that you can compress a many-pixeled test face image (pick one from the test image array) using a small set of image vectors (e.g., 10, 20, or 50).

Before you dig into this problem, think through how you would formalize the face data compression as a problem that you can solve with the linear algebra techniques that you've learned so far. There is no exercise to answer, but we want you to think through these steps before going further in the assignment.

- How you will choose your set of image vectors?
- Come up with the steps needed to do the above and write some pseudo code (e.g., load the data, vectorize the images, etc.).
- How could you tell whether your compression algorithm works (these could either be quantitative metrics, like root-mean squared error or qualitative metrics).

14.3 Eigenfaces for Face Recognition

It's time to bring it all together and finally plunge into the prosopagnosia (look it up) problem (or at least build some facial recognition software, but alliteration is fun). You will implement the eigenfaces algorithm to identify photos of your classmates. While it sounds fancy, you have almost all of the pieces needed to understand and implement Eigenfaces (the last necessary piece you will pick up momentarily). Here are the major steps in the Eigenfaces algorithm.

1. Use PCA to compute the k principal components of the training face images (the k eigenvectors with largest eigenvalues).
2. Project the training and test face images onto the k principal components. We'll call this the *facespace* representation of our original images.
3. For each of the test images, compute the closest match between the test image (represented as a k-dimensional vector in facespace) and the training images (again, in facespace). The notion of "closest match" here can be described in a few different ways, but the easiest thing to do is to use

the the Euclidean distance to define how far apart two points are. In this way, you would look for the training point that has the smallest Euclidean distance for a particular test point and predict the identity of the test point to be the same as the identity of this closest training point. This method of classification is known as nearest neighbor classification and it is the one new concept you need to implement Eigenfaces.

Exercise 14.4

Earlier in this assignment you wrote some pseudo code for face compression, which as you can see from the description of Eigenfaces above, gets you most of the way there. Before you actually implement Eigenfaces, we'd like you to extend your pseudocode to cover the whole Eigenfaces algorithm. In addition to the steps of Eigenfaces describe above, you should also think about the steps needed to calculate the accuracy of your system (i.e., how often does it get the person's identity correct).

Exercise 14.5

Implement the eigenfaces algorithm.

1. Your code, which can be a script, function, or livescript, should use the training and test sets of images provided: `classdata_train.mat` and `classdata_test.mat`, respectively.
2. You may want to start by identifying one face from the test set, but by the time you are done, your code should run through all of the test images and report the fraction it guesses correctly.
3. Test different numbers of eigenvectors. How many does it take to guess right most of the time?
4. If you'd like, time how long it takes your code to run. Can you do anything more efficiently to make it run faster? (use `tic` and `toc` in MATLAB for timing)
5. Visualize the first few eigenfaces. Can you interpret what they mean?
6. Generate a figure that depicts the success rate (accuracy at determining the identify of a person in an image) versus the number of eigenfaces used.
7. Generate a figure that depicts the success rate (accuracy at determining the identify of a person in an image) versus the number of eigenfaces used where the training data consists of only images of people not smiling and the test data consists of only images of people smiling (these are located in `classdata_non_smiles.mat` and `classdata_smiles.mat` respectively. Comment on the difference in performance on when using these files versus the data from the previous parts of this problem.

Guidelines:

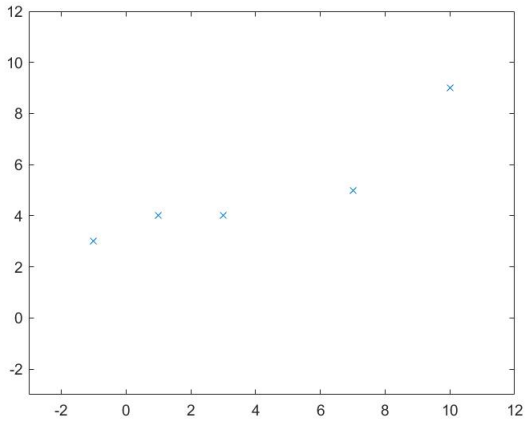
- You should comment your code (use %) so others could read and understand it.
- Don't use the command `pca`, but instead build your algorithm using either the `eig` or `eigs` command (`eigs` computes just a few eigenvectors, which can be faster when you only care about the eigenvectors with large eigenvalues). We want you to think through all the steps involved in your facial recognition program, and that means doing the math "yourself".

(Optional) Extensions (These are not spelled out in much detail. We recommend you talk to a member of the teaching team before trying these (especially the second two).

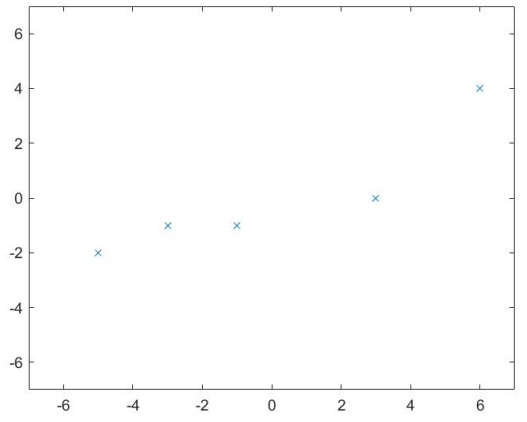
- Analyze the mistakes your algorithm makes (particularly when training on non-smiles and testing on smiles).
- Use Eigenfaces to do smile detection instead of identity recognition.
- Combine Eigenfaces with a classifier other than nearest neighbors (e.g., formulate an LSAE to create a series of one person versus everyone else detectors).
- Get your system working on live video.

Solution 14.1

1.
```
>> D = [-1 3; 1 4; 3 4; 7 5; 10 9]
>> plot(D(:,1),D(:,2),'x')
>> axis ([-3 12 -3 12])
```

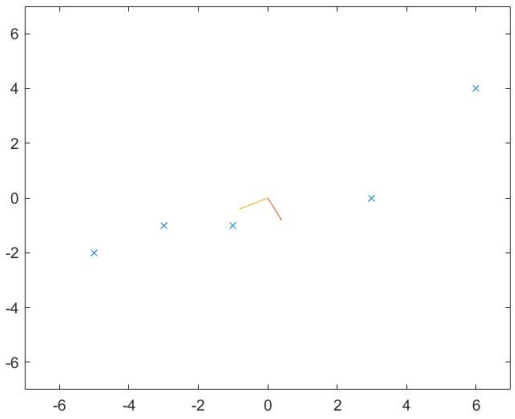


2.
```
>> mumatrix=[mean(D(:,1))*ones(5,1) mean(D(:,2))*ones(5,1)]
>> tildeD=D-mumatrix
>> plot(tildeD(:,1),tildeD(:,2),'x')
>> axis ([-7 7 -7 7])
```



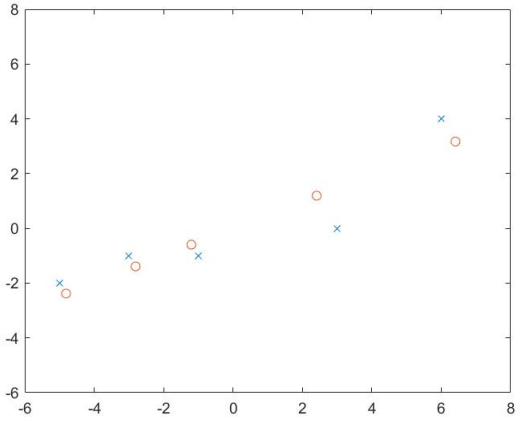
3.
```
>> [Vec,Diam]=eig(tildeD'*tildeD)
```

```
>> hold on
>> quiver(0,0,Vec(1,1),Vec(2,1))
>> quiver(0,0,Vec(1,2),Vec(2,2))
```



4.

```
>> proj=tildeD*Vec(:,2)
>> B=proj*Vec(:,2)'
>> hold on
>> plot(tildeD(:,1),tildeD(:,2),'x')
>> plot(B(:,1),B(:,2),'o')
>> axis ([-6 8 -6 8])
```



Solution 14.2

1. No. If we write a data point $a\mathbf{p}_1 + b\mathbf{p}_2$, then the reduced dimension version is $b\mathbf{p}_2$. It's impossible to recover a, which is the information in the perpendicular direction.
2. We would get the information in the perpendicular direction, which we can interpret as the "error" in reducing the dimension from $\mathbf{D}$ to $\mathbf{B}$.
3. You can use the error $\mathbf{B} - \mathbf{D}$.
4. We can write a new data point $\mathbf{d}$ as

$$(\mathbf{d} \cdot \mathbf{p}_1)\mathbf{p}_1 + (\mathbf{d} \cdot \mathbf{p}_2)\mathbf{p}_2.$$

Solution 14.3

1. ```
   >> A = (1/sqrt(7304))*[b_tr-mean(b_tr) w_tr-mean(w_tr) s_tr-mean(s_tr)];
   >> R=A'*A
   ```

2. ```
   >> [V,D]=eig(R)
   >> Vp=[V(:,2) V(:,3)]
   ```

3. ```
   >> T = [b_new-mean(b_new) w_new-mean(w_new) s_new-mean(s_new)]'
   ```

4. ```
   >> alpha=Vp'*T;
   ```

Solution 14.5

A reference implementation of Eigenfaces is linked from the Canvas assignment page.

Chapter 15

Day 8: Eigenface Synthesis and Project Kick-Off

Schedule

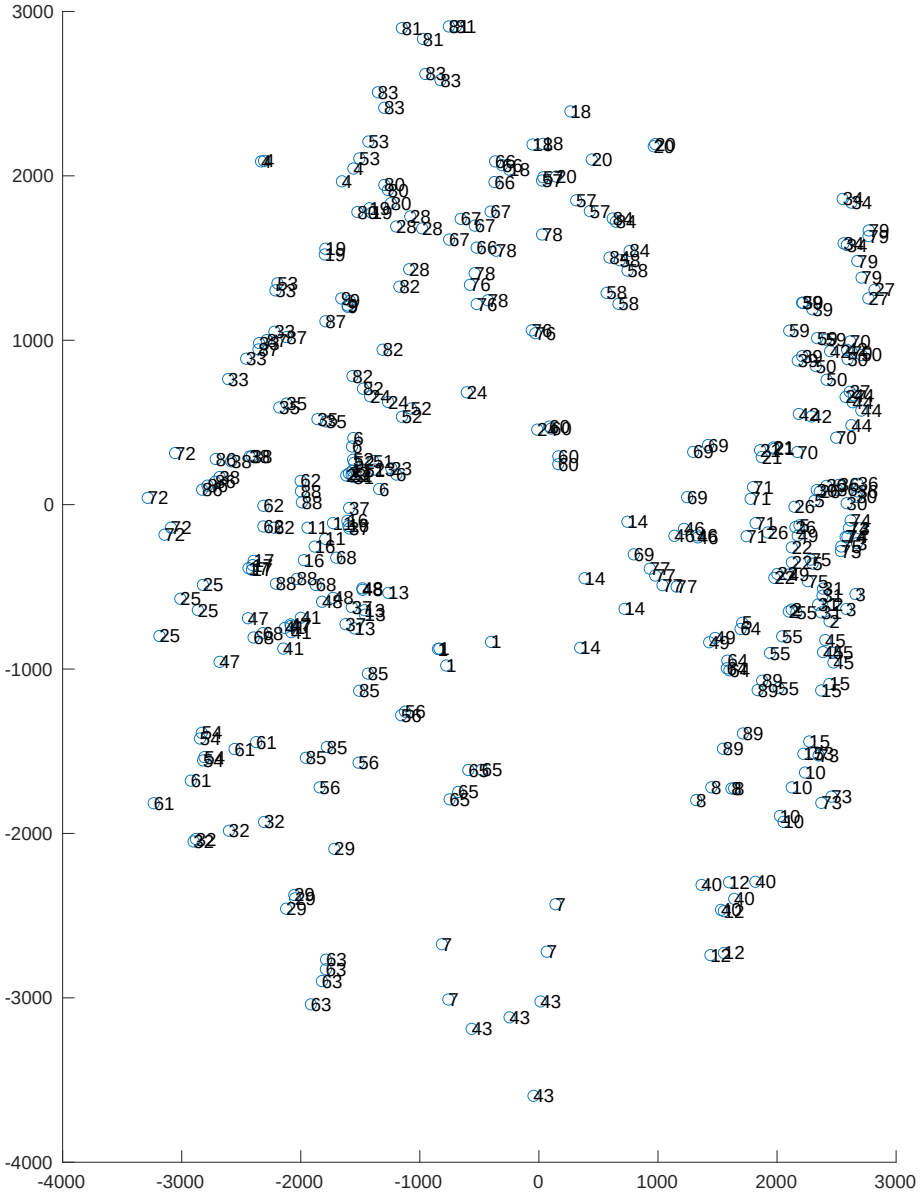
- 0900–1015 Debrief and Synthesis Activity on Eigenfaces
- 1015–1030 Coffee Break
- 1030–1115 Singular Value Decomposition — Theoretical
- 1115–1200 Singular Value Decomposition — In action
- 1200–1220 Review and Preview
- 1220–1230 Survey

15.1 Debrief and Synthesis Activity on Eigenfaces

During the overnight assignment you implemented a facial recognition routine. For the first part of the morning, we want you to work with your table mates to think again about your method and implementation.

An Aside on Nearest Neighbor Classification

One area where folks ran into trouble on the night assignment was the nearest neighbor approach to classification. In nearest neighbor classification you classify a test point, $\mathbf{x_t}$ (think of representing this as a column vector containing a new piece of data like a face image of someone whose identity we don't know), by comparing it to a set of labeled training points $(\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), \ldots, (\mathbf{x_n}, y_n)$. Each y_i is the thing we are trying to predict (e.g., the identify of the person in a face image). In nearest neighbor classification we check to see which of the n training points is closest to the test point. When we find the one that is closest, we predict the label for $\mathbf{x_t}$ to be the same as the label of the closest point in the training set. To help get the idea across, here is a figure that shows all of your faces projected onto the first two principal components. Also shown are the subject ids for each point. You should be able to see from the figure that points with the same subject id tend to cluster together (i.e., are close together in facespace). This is where the power of nearest neighbor classification comes from.


3000
2000
1000
0
-1000
-2000
-3000
-4000
-4000
-3000
-2000
-1000
0
1000
2000
3000

Eigenfaces Review [45 mins]

We'd like you to take this opportunity to discuss with your table-mates the approach you took to facial recognition and the way you implemented it in MATLAB. The goal here is to think through the method at different levels from conceptual to code, resolving any confusion, and identifying whether any issues are primarily at the conceptual-level, the implementation-level, or the translation space in between. **We highly recommend that you set aside your existing code and focus on developing the approach with your table-mates.** Here are some questions to guide you:

- Will you pre-process the data? If so, how?
- How will you compute the Eigenfaces?
- How will you decide how many Eigenfaces to include? Which Eigenfaces might you leave out?
- How will you identify a face?
- How will you measure the accuracy of your implementation?
- How will you handle false positives?

Exercise 15.1

1. Sketch out on the board your conceptual approach to facial recognition—use words and diagrams here, and think of a set of key frames.
2. Sketch out on the board your mathematical approach to facial recognition—translate the key frames from your conceptual approach to mathematical expressions or equations.
3. Sketch out on the board your implementation approach to facial recognition—translate the key conceptual and mathematical ideas to MATLAB pseudo-code.

Eigenfaces Paper [45 mins]

This is on the next overnight assignment, but it would make sense to start this now if you have the time.

Check out *Eigenfaces for Recognition*, an early paper on eigenfaces, by M. Turk and A. Pentland. You have most of the tools to understand this paper, but the writing style might be unfamiliar (intense!). Spend 1 hour on this paper and then feel free to move on. The first 6 pages of this paper describe the use of eigenfaces in face recognition.

Check out other sources as well. Wikipedia is pretty useful for eigenfaces, and this later paper talks about eigenfaces and an extension called Fisherfaces (not fish faces).

Exercise 15.2

We are asking you read this paper for several reasons. We hope that it highlights and synthesizes all the material you've learned in this module. It will also give you practice reading a technical paper, which is a skill you'll continue to develop over your career.

1. In what ways was your approach to implementing the eigenfaces algorithm similar or different from the authors' approach?
2. In what ways did your understanding of the eigenfaces algorithm change after reading the paper?
3. Were there places in the reading that you "got stuck?" If so, how did you address that?
4. What questions do you have after reading the paper?

15.2 Singular Value Decomposition (SVD) — Theoretical

We previously met the Eigenvalue Decomposition (EVD), which we used on square matrices. There is no EVD for rectangular matrices, but there does exist a generalization known as the Singular Value Decomposition, which is one of the most useful matrix decompositions in applied linear algebra. We will begin with some background concepts on singular values and singular vectors, and then explore them in the context of a user-movie rating data matrix. See the following webpage at the American Mathematical Society for a good geometric discussion of the SVD.

The Big Idea

Rectangular matrices don't have eigenvalues and eigenvectors. However, they have a generalisation of these known as singular values and singular vectors.

- The singular values σ_i and singular vectors $\mathbf{u}_i, \mathbf{v}_i$ of an $n \times m$ rectangular matrix $\mathbf{A}$ satisfy the definition

$$\mathbf{A}\mathbf{v}_i = \sigma_i \mathbf{u}_i \tag{15.1}$$
$$\mathbf{A}^T\mathbf{u}_i = \sigma_i \mathbf{v}_i \tag{15.2}$$

  The singular vectors $\mathbf{v}_i$ are known as the **right singular vectors** and the singular vectors $\mathbf{u}_i$ are known as the **left singular vectors**.

- There are precisely $r = min(n, m)$ non-zero singular values. The singular vectors $\mathbf{v}_i$ are the eigenvectors of $\mathbf{A}^T\mathbf{A}$, and the singular vectors $\mathbf{u}_i$ are the eigenvectors of $\mathbf{A}\mathbf{A}^T$. The r non-zero eigenvalues of $\mathbf{A}^T\mathbf{A}$ and $\mathbf{A}\mathbf{A}^T$ are σ_i^2.

- The $n \times m$ matrix $\mathbf{A}$ has a *singular value decomposition* (SVD) of the form

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{15.3}$$

  where $\mathbf{U}$ is an $n \times r$ orthogonal matrix whose columns are $\mathbf{u}_i$, $\mathbf{\Sigma}$ is an $r \times r$ diagonal matrix with r non-zero entries σ_i, and $\mathbf{V}$ is an $m \times r$ orthogonal matrix whose columns are $\mathbf{v}_i$. Please note that

this version of the SVD is called the *reduced* or *economy* SVD - there is a more general form but this is the most useful in a practical setting.

Exercise 15.3

1. Read "The Big Idea" again!
2. Let's assume that $\mathbf{A}$ is a 3×2 matrix. What is the size of $\mathbf{A}^T$? What is the size of $\mathbf{v}_i$ and $\mathbf{u}_i$? What is the size of $\mathbf{A}^T\mathbf{A}$ and $\mathbf{A}\mathbf{A}^T$? How many eigenvalues will $\mathbf{A}^T\mathbf{A}$ have? How many eigenvalues will $\mathbf{A}\mathbf{A}^T$ have? What must be true about these eigenvalues according to "The Big Idea"?
3. Show that σ_i^2 and $\mathbf{v}_i$ are the eigenvalues and eigenvectors of $\mathbf{A}^T\mathbf{A}$ by multiplying Equation (15.1) by $\mathbf{A}^T$ and then using Equation (15.2) to simplify.
4. Show that σ_i^2 and $\mathbf{u}_i$ are the eigenvalues and eigenvectors of $\mathbf{A}\mathbf{A}^T$ by multiplying Equation (15.2) by $\mathbf{A}$ and then using Equation (15.1) to simplify.
5. Take the transpose of Equation (15.2) and justify the use of the term **left singular vector** for $\mathbf{u}_i$.
6. Why is it valid to write

$$\mathbf{A}[\mathbf{v}_1 \dots \mathbf{v}_r] = [\mathbf{u}_1 \dots \mathbf{u}_r] \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_3 \end{bmatrix}$$

   and why does this imply Equation (15.3)?
7. Why does Equation (15.3) imply that

$$\mathbf{A} = \sigma_1\mathbf{u}_1\mathbf{v}_1^T + \sigma_2\mathbf{u}_2\mathbf{v}_2^T + \dots + \sigma_r\mathbf{u}_r\mathbf{v}_r^T \tag{15.4}$$

8. An $n \times m$ matrix has nm data values. How many data values do you need to store σ_1, $\mathbf{u}_1$, and $\mathbf{v}_1$? What kind of compression ratio would you have if you only stored the first singular value and the first singular vectors?

15.3 *Singular Value Decomposition (SVD) — In Action*

We're going to be using a LiveScript notebook to go through an example of using SVD to analyze movie ratings.

You'll need to download both the dataset (movielens25m.mat) and the LiveScript (movieLens25m.mlx) before getting started. We have included a PDF of the LiveScript in this document for your referene.

Analyzing Movie Ratings via SVD

In this notebook we're going to be working with a subset the MovieLens25M dataset. The original dataset (https://grouplens.org/datasets/movielens/25m/) contains

- 25 million ratings
- from 162,000 users
- on 62,000 movies

The dataset was generated on November 21st, 2019, so it is pretty current. In this activity, we're going to be using a reduced subset of this data that only includes popular movies (that have been rated at least 1000 times) and users that have rated lots of movies (at least 500). This leaves us with

- 7.1 million ratings
- from 9,663 users
- on 3,790 movies

The goals of this activity are threefold.

1. To work with a different type of data than images or temperatures (here we will be working with ratings). Applying the tools you have learned in this module to different domains will help solidify your learning, help you see connections, and potentially get you excited for your module 1 project.
2. To see how SVD can be used to examine the important trends in your data (since we had lots of practice with using the EVD on the overnight).
3. To have some fun!

To get started, we're going to load the data and display a little bit of the data. Please see the comments in the code for some more information.

```
load('movielens25m.mat');
sizeOfMovies = size(movies)
% the cell array `movies` is 3706 by 3.  Each of the 3706 entries correspopnds to a
% particular movie, and along the second dimension the entries correspond to the movie
% the movie title, and the movie genre
%
% Here we extract the information about the first movie in the dataset
[movieId, movieTitle, movieGenre] = movies{1,:}

ratingsSize = size(ratings)
% the matrix `ratings` is 6040 by 3706 and encodes the rating that a
% particular user (row) gave to a particular movie (column).  The ratings
% are 1, 2, 3, 4, or 5 stars or the special value NaN (not a number) if the
% user didn't rate that particular movie.
%
% Let's look at the ratings that were given to the first movie in the
% dataset, which as we saw is Toy Story.  We can do this using the histc
% function (we'll ignore missing values in this analysis)
possibleRatings = [0.5:0.5:5];
nRatings = histc(ratings(1,:), possibleRatings);
figure;
```

```
bar(possibleRatings, nRatings);
xlabel('Rating');
ylabel('Number of Users');
title(['Ratings for ', movieTitle])
```

Okay, yeah that was a pretty great movie. Let's check out a less good movie, Anaconda. Highly recommended!! Look at this cast https://www.imdb.com/title/tt0118615/fullcredits !!!

```
anacondaIndex = 850;
[movieId, movieTitle, movieGenre] = movies{anacondaIndex,:}
nRatings = histc(ratings(anacondaIndex,:), possibleRatings);
figure;
bar(possibleRatings, nRatings);
xlabel('Rating');
ylabel('Number of Users');
title(['Ratings for ', movieTitle])
```

Cleaning up the Data

As you probably guessed, we're going to be applying SVD to this data. Before we start analyzing this data, we're going to do a few things to make the problem a bit easier to handle. First we're going to have to deal with the fact that we have a bunch of missing values in our ratings matrix (i.e., movies that particular users did not rate). The step of filling in missing values is called **data imputation**. There are many ways to do this, but we've chosen a particularly easy strategy of simply replacing any ratings with the average rating of that particular movie (e.g., if a user didn't rate Toy Story, we would fill it in with the average rating of Toy Story based on the other users in the dataset who actually rated that movie).

```
ratingsFilled = fillmissing(ratings, 'constant', nanmean(ratings));
```

As a final data cleaning step, we're going to subtract out the mean of each row. This will control for the fact that users vary considerably in how the numerical score they assign to movies (e.g., one user's 3 may be more comparable to amother user's 1).

```
ratingsMeanCentered = ratingsFilled - mean(ratingsFilled,2);
```

Framing the Problem Using SVD

Next, let's think about how SVD might help us to analyze this dataset. Suppose we compute the SVD of the matrix `ratingsMeanCentered`. Let's use $\mathbf{u_1}$ to refer to the first left singular vector, $\mathbf{v_1}$ to refer to the first right singular vector, and σ_1 to refer to the first singular value (let's assume that the first pair of singular vectors has the largest singular vector).

Exercise

Before running any other code in this notebook, answer the following questions regarding the first pair of singular vectors.

1. What are the sizes of $\mathbf{u_1}$ and $\mathbf{v_1}$? What do each of the dimensions of $\mathbf{u_1}$ correspond to? How about each dimension of $\mathbf{v_1}$?

2. In 15.3.8 we talked about compressing the original matrix down to m + n + 1 values. If we think of $\mathbf{u_1}, \mathbf{v_1}, \sigma_1$ as the compressed version of ratings data, how would we reconstruct the ratings data using $\mathbf{u_1}, \mathbf{v_1}, \sigma_1$ (you essentially did this already, we're hoping you can recall this fact from earlier and apply it here).
3. We can think of $\mathbf{v_1}$ as encoding the dominant trend that explains the ratings of each movie. For this dataset, what might this correspond to?
4. We can think of $\mathbf{u_1}$ as encoding the dominant trend that explains the ratings by each usere. For this dataset, what might this correspond to? Keep in mind we have already subtracted out the mean of each user. It might be helpful to expand your formula from problem 2 to see how $\mathbf{u_1}, \mathbf{v_1}, \sigma_1$ interact with each other.

Now we're going to compute the SVD. We'll just compute the 10 pairs of left and right singular vectors with the largest singular values.

```
[U, Sigma, V] = svds(ratingsMeanCentered, 10);
```

Examining the Right Singular Vectors

Now that we've computed our singular vectors, let's see if we can make sense of them. It turns out that the right singular vectors (the ones that have to do with movies) are generally more interpretable than the left singular vectors (the ones that have to do with users). We'll start out by looking at each right singular vector.

Exercise

Before running the code, think through the following question with your table-mates.

What might you do in order to make sense of what a particular right singular vector represents? Consider things like examining small or large values, looking for correlations, etc. There's not only one right answer, so throw out some ideas and try to think through what examining a particular aspect of the vector might tell you.

(we'll leave a little space to make it easier not to look at what we did)

Looking at Large and Small Values

One simple way to understand the right singular vectors is to look at the largest and smallest components of each vector. This will tell us which movies are either most strongly (positively) and most strongly (negatively) associated with this component. In the code below, we'll print out the title, genre, and component of the 10 movies that are most positively and most negatively associated with each right singular vector. **Exercise: Based on these outputs, can you tell a story about what the singular vector represents?**

```
for i = 1 : 10
    disp(['Component ', num2str(i)]);
```

```
    getHighAndLowMovies(V(:,i), movies)
end
```

Examining the Left Singular Vectors

Now we're going to check out the left singular vectors.

Exercise

Before running the code, think through the following question with your table-mates.

What might you do in order to make sense of what a particular left singular vector represents? Consider things like examining small or large values, looking for correlations, etc. There's not only one right answer, so throw out some ideas and try to think through what examining a particular aspect of the vector might tell you.

(we'll leave a little space to make it easier not to look at what we did)

Looking at Large and Small Values

Similarly to what we did for the right singular vectors, let's take a look at large (positive) and small (negative) components of each singular vector. Instead of looking at the top 10 and bottom 10, we're instead going to look at the single highest and single lowest component (each of which correspond to a user). For that user, we're going to show a sampling of movies that the user rated (focusing on the top 10 and bottom 10 ratings for that particular user). **Exercise: Given what you know about the corresponding right singular vector, try to make sense of the users that are at either extreme of the left singular vectors.**

```
for i = 1 : 10
    [~, highestUserIndex] = max(U(:,i));
    [~, lowestUserIndex] = min(U(:,i));
    disp('');
    disp(['Component ', num2str(i)]);
    disp('The user with the largest component rated the following movies as high and lo
    getHighAndLowUserRatings(highestUserIndex, movies, ratings)
    disp('The user with the smallest (probably negative) component rated the following
    getHighAndLowUserRatings(lowestUserIndex, movies, ratings)
end
```

Next Steps

To give you a sense of where you might take this in a project, here are some things you might investigate next with this dataset.

1. We didn't really look at how you would use the SVD to make recommendations. It turns out the SVD can be used to come up with good guesses for the missing values in the original ratings matrix (the NaNs) and you can then provide recommendations based tailored for a praticular user.
2. We didn't quantify how well the svd predicted the ratings. In order to do that, you could divide the ratings into a training and test set and see how well your SVD model can predict the test ratings (i.e., a rating set that wasn't used to compute the SVD).
3. We filled in the missing values with the means of each movie, but there are variants of SVD that can handle the missing values directly (they do entail tradeoffs). You could inverstigate how one of those methods would work on this data.

```
function movieExtremes = getHighAndLowMovies(v, movies)
    % return a cell array with the most positive and most negative
    % components of the right singular vector v.
    nHighLow = 10;
    movieExtremes = cell(nHighLow*2, 3);
    [c, indices] = sort(v);
    movieExtremes(1:nHighLow,1) = movies(indices(end-(nHighLow-1):end),2);
    movieExtremes(1:nHighLow,2) = movies(indices(end-(nHighLow-1):end),3);
    movieExtremes(1:nHighLow,3) = num2cell(c(end-(nHighLow-1):end));
    movieExtremes(1+nHighLow:end,1) = movies(indices(1:nHighLow),2);
    movieExtremes(1+nHighLow:end,2) = movies(indices(1:nHighLow),3);
    movieExtremes(1+nHighLow:end,3) = num2cell(c(1:nHighLow));
end

function userRatings = getHighAndLowUserRatings(userIndex, movies, ratings)
    % return a cell array with the most positive and most negative reviews
    % given by the specified user
    nHighLow = 10;
    userRatings = cell(nHighLow*2,2);
    [r, indices] = sort(ratings(userIndex,:));
    % filter out NaNs
    indices = indices(~isnan(r));
    r = r(~isnan(r));
    userRatings(1:nHighLow,1) = movies(indices(end-(nHighLow-1):end),2);
    userRatings(1:nHighLow,2) = num2cell(r(end-(nHighLow-1):end));
    userRatings(1+nHighLow:end,1) = movies(indices(1:nHighLow),2);
    userRatings(1+nHighLow:end,2) = num2cell(r(1:nHighLow));
end
```

Chapter 16

Night 8

16.1 Eigenfaces Paper

Check out *Eigenfaces for Recognition*, an early paper on eigenfaces, by M. Turk and A. Pentland. You have most of the tools to understand this paper, but the writing style might be unfamiliar (intense!). Spend 1 hour on this paper and then feel free to move on. The first 6 pages of this paper describe the use of eigenfaces in face recognition. Check out other sources as well. Wikipedia is pretty useful for eigenfaces, and this later paper talks about eigenfaces and an extension called Fisherfaces (not fish faces).

Exercise 16.1

We are asking you read this paper for several reasons. We hope that it highlights and synthesizes all the material you've learned in this module. It will also give you practice reading a technical paper, which is a skill you'll continue to develop over your career.

1. In what ways was your approach to implementing the eigenfaces algorithm similar or different from the authors' approach?
2. In what ways did your understanding of the eigenfaces algorithm change after reading the paper?
3. Were there places in the reading that you "got stuck?" If so, how did you address that?
4. What questions do you have after reading the paper?

16.2 Beginning the Project

In this project you will extend the work you have already done on facial recognition and feature detection by analyzing the performance of an existing algorithm within a real context. We know that facial recognition and other forms of feature detection algorithms are incredibly powerful, but they are often prone to failure, and those failures can have very real consequences on people's lives. This is a new formulation of this project where we are challenging you to think deeply about the contexts and consequences for facial/feature recognition algorithms. To prepare for tomorrow's in-class ideation activities, we ask you to do two things:

1. Read the project description, which can be found in the next chapter (Chapter 17), and write down any questions you find yourself asking.
2. Fill out this partner survey by midnight tonight, Feb. 19th. We will announce the teams tomorrow morning.

Chapter 17

Faces Project: The Context and Consequences of Feature Recognition, Detection, and Classification

17.1 Overview

This is a project that asks you to extend the work you have already done on facial recognition and feature detection by analyzing the performance and considering the consequences of an existing algorithm within a real(ish) context. This is a fairly new formulation of this project and we are giving you the freedom (and responsibility) to choose an interesting path and follow it judiciously.

The LinAlgCo owns the rights to all uses of linear algebra. They've recently become aware of the use of linear algebra in *feature recognition, detection, and classification* ("*FeaRDeClass*") algorithms. The company is concerned about the ethical implications of the widespread use of these algorithms. They don't want to tarnish the good name of linear algebra. You have been hired as a consultant to address these concerns.

Specifically, they've asked you to do the following:

1. Identify a specific context in which linear algebra is being used to do *FeaRDeClass*.
   (Examples: Smile detection using linear regression, identifying missing persons from photos in social media, unlocking your phone with your face using Eigenfaces, classifying movie preferences...)
2. Pose a question about the context in which *FeaRDeClass* is being preformed and the possible ethical implications. You do not have to be able to answer this question, but it should guide your investigations.
   (Examples: Privacy, bias, misuse, reinforcing negative structures...)
3. Answer some part of your question by analyzing the results of a *FeaRDeClass* algorithm. You can use any *FeaRDeClass* algorithm as long as you can explain how it works. You need to do some quantitative analysis, but the specifics are up to you.
   (Examples: Does the facial recognition have higher accuracy with group X than group Y? Are STEM documentaries more likely to be recommended to men than women?)

Your consulting team is expected to produce a formal report, due to LinAlgCo by Monday March 2nd at 9:00am.

17.2 What we expect you to do

1. Start with some background research on contexts for *FeaRDeClass* and their associated ethical issues. This research will help you to choose what to focus on, and you should also reference this research when discussing the context of your project in the introduction of of your report.

2. Choose an important question related the context and ethics of your chosen *FeaRDeClass*. This should be rooted in a real context, and you do not have to be able to answer it. Break off a small sub-question that you think you can answer in one and a half weeks through an analytical approach that utilizes eigenfaces, another facial recognition algorithm, or linear regression. Consider the ethical consequences of your chosen topic. Under what circumstances could the technology be harmful? Whom might it harm, and how?

3. Plan, execute, and document some analysis (which could include modifying/creating an algorithm) to answer your sub-question.

4. Explain the mathematical algorithm you are using in detail, explaining the various steps and what the purpose of each step of the process is. Use equations!

5. Explain how the results of your analysis inform the question you are trying to answer. Tie the results of your sub-question back into your larger question and chosen context. What can you conclude from the analysis you did? Recommend areas for future investigation.

6. You should understand the metrics against which your programs should be measured. How do you characterize the accuracy of your approach? Against what should you compare this accuracy? How do you quantify the consequences of your approach?

7. Communicate the context, analytical approach, and findings via a formal technical report to the LinAlgCo.

17.3 Resources

1. Your existing eigenfaces algorithm or the example solution posted. Let us know if you need help getting eigenfaces working.

2. The smile detection algorithm, which uses linear regression. Your version or the walkthrough from class can be modified to do something similar.

3. Training and test images for your class and past QEA classes.

4. The 10k faces database. This includes >2,000 images that have been classified in terms of demographics and other info (like whether people are facing the camera) and a software tool to narrow the database by classifiers (e.g., to only smiling men). The downside of this database is there is only one photo of each person.

5. The internet. In addition to doing context/background research, you can go find a different algorithm or face database if you prefer, but be aware that this will take extra time!

6. Your teaching team. Remember that we are here to support your learning! Bounce ideas off of us in office hours. Don't let MATLAB get you down; ask for help early and often.

17.4 Deliverable

You need to produce a written report, but we've broken it into a few sub-assignments to keep you on track and create opportunities for feedback.

1. Due Monday 2/24: An informal document outlining your chosen context, big question, sub-question, and the algorithm you will use to answer your sub-question, plus a plan for what analysis you will do and what kind of results you will get. (You should also get started on the analysis before Monday, but you don't have to turn any results in yet.) This document should serve as an outline for your final report.
2. Due Thursday 4/27: A draft of your written report. The teaching team will give you feedback. The more complete the draft, the more helpful your feedback will be! You should bring a printed version to class.
3. Due Monday 3/2: A final version of your written report.

17.5 Project report

You will generate a professional-looking and edited report to send to LinAlgCo that summarizes and justifies the decisions you have made. The executives at LinAlgCo are familiar with linear algebra and mathematics notation, but you should not assume they know anything about *FeaRDeClass* algorithms in general or the particular one you are studying.

The goal of the report should be to help LinAlgCo understand the context and consequences of the specific *FeaRDeClass* algorithm you have questioned and analyzed. You are NOT writing a story about what you did in the project. Aim for content and clarity, not length.

Structure

The report should have the following six sections (and you might want to break them into subsections):

1. Summary

What will I find in this report?

Open with a one paragraph summary that orients LinAlgCo to what they will find in the report. It should make clear why the report was written, what each section will accomplish, and what the key insights and results are. You should also summarize your recommendations.

2. Introduction

What is this project?

Your introduction should: (1) Provide the background and context for the *FeaRDeClass* that you have chosen to analyze. When and where is it used? By whom? What are the general technological or social issues associated with its use? (2) Explain your algorithm technically. How does it work? Bear in mind your audience. (3) Lay out the general ethical implications of the algorithm that you are investigating. Under what circumstances could the technology be helpful or harmful? Whom might it help or harm, and how? (4) Clearly state, within the broader ethical context, what question or issue you are exploring and and what sub-question you are quantitatively investigating.

3. Methods

How does your approach work, and what did you do with it?

Having introduced the reader to terminology and ideas, this section should lay out the approaches you are using, both in terms of the chosen algorithm and the analysis you are doing with it. Use equations and define all variables.

4. Detailed Findings

What are the main results and consequences of your work?

This section should contain your main results and consequences of your work which you have quantified. This section should contain some clear, informative, labeled, and captioned plots and images that demonstrate your findings. Quantitative results should be clearly connected to the context of the investigation. Why are your findings meaningful? Reflect on the downsides of the technology and the people it could hurt, and suggest some strategies for improvement.

5. Recommendations

What are the key takeaways? Summarize the key findings of the report, situate them in the greater context, and identify areas for future investigation. This section should be concise—they details go in the previous section.

6. References

Provide full citations for sources referenced in the paper. Format doesn't matter here as long as you provide sufficient information about each of your sources.

17.6 Grading rubric

1 pt. Summary presents a clear, high-level overview of paper.

3 pts. Question being investigated is clearly rooted in a real context, as discussed in the introduction and justified with references. Discussion of potential for harm is thorough.

2 pts. Algorithm is clearly explained using equations and words.

2 pts. Analytical approach is clearly explained.

2 pts. Findings are justified with appropriate figures and discussion.

2 pts. A clear connection is drawn between the findings and the original sub-question, greater question/issue, and greater context.

2 pts. Paper is logically organized and writing, figures, and equations are polished.

Chapter 18

Day 9

Schedule

- 0900–0930 Debrief Eigenfaces Paper
- 0930–1030 Project Ideation
- 1030–1045 Coffee Break
- 1045–1225 Project Worktime

18.1 Debrief Eigenfaces Paper

One of the goals of QEA is to develop your abilty to read technical papers and implement the main ideas. Much of engineering practice is based on building on the incredible work of those who came before us, and being able to critically read technical writing is a skill to be developed. Similarly, identifying the particularly helpful, or challenging, portions of a paper can help you improve your own technical writing. Please consider the following prompts:

Exercise 18.1

1. What did the **authors set out to show**? Why did the authors choose this particular goal?
2. What were the **best parts** of the paper? Was the approach to facial recognition easy to follow? What did the authors do particularly well?
3. Identify the **missing pieces** in the paper. These could be missing analyses, missing punch-line graphs, missing equations, etc. Another way to think about this is which of the main conclusions of the paper are you still the most skeptical of. Your skepticism could arise from a lack of evidence, or because the evidence was not presented in a clear and easily digestible format.

18.2 Project Ideation

You should be seated with your partner for the rest of class.

User Ideation Extravaganza

There are A LOT of possible questions you could propose for this project. In the project document we prompt you to *choose an important question related to feature recognition, detection, or classification.* This should be rooted in a real context, and you will likely not be able to answer it entirely. Break off a small subquestion that you think you can answer in one week through an analytical approach that utilizes eigenfaces, another

facial recognition algorithm, or linear regression. We recognize that this is a very open ended and somewhat ambiguous prompt, but we feel you are up to the challenge! We will be taking the remainder of class to generate lots of possible questions, refine ideas, and try to generate teams around shared interests.

Exercise 18.2

1. Do this part individually [10 minutes]: Write down as many different ideas for possible questions related to facial/feature recognition/detection/classification as you can on different sticky notes. Go wild!
2. Do this part with your table and the neighboring table [10 minutes]: Our goal now is to get all of the questions up on the back board and to group them (in other words, all of the questions around flagging and the potential bias in that process could go into one group). Draw on the board as needed. This may feel a bit chaotic but we will help you get there.
3. Do this part with your partner [5 minutes]: Walk around the room with your partner, reading the idea clusters from the other table groups. Get inspired!

Pair Project Ideation

This next stage is going to give you an opportunity to work with your partner to develop a complete project idea and to "pitch" it to the class.

Exercise 18.3

1. With a partner at your table, select a question (or group of questions) from the boards. Grab the appropriate sticky notes and bring them to your table. [5 minutes]
2. Collaboratively develop an idea that identifies a question, it's context, and how you could perform some analysis. Do some internet research to find out more about your question and its context. What are the ethical issues associated with your topic? What is known and what is still in question? Fill out the project pitch handout. You will need to define the question itself, the real world context, and details about the critical concepts from this module that will allow you to perform the desired analysis. [20 minutes]

18.3 Project Worktime

Get started on the details of your project. You should consider this an extension of the project ideation time. Play with ideas, and hopefully, by the end of class you'll feel like you've settled on an idea and have a direction to go in. Use this time to chat with the faculty.

Chapter 19

Night 9

Project work time. Remember that the outline of your report is due Monday. You should also get started on the analysis before Monday, but you don't have to turn in any results yet. The last few sections can simply be an outline of what you will do.

Chapter 20

Day 10

Project work time.