Module One: The Basics

Eric Johnson and Madhav Mani

May 11, 2020

Before we can really talk about building and assessing statistical models with data, you will need to be familiar with some basics of probability theory, coding, and simulation. This module will provide these basics as well as motivate why our framework for confronting data from a statistical viewpoint is justified. You will quickly see that while some of these basics resemble what you might learn in an introductory statistics textbook, our emphasis will reside much more on the practical and intuitive aspects of probability rather than on developing a series of rules and tests. In particular, we believe that a modern quantitative thinker knows not just the **theory** of what a probability density function is (for example), but also how to **generate** one from data or simulations, how to **visualize** one, and how to **manipulate** one to make different calculations.

These notes have three sections: the first section will simply link to the Python tutorial that should be completed before/during the reading of this text. The next two sections will then introduce basic probability concepts, such as probability and frequency distributions, and Bayes Theorem, respectively. Each section will emphasize **theory, calculation, visualization**, and **simulation** somewhat evenly. At the end of the chapter will be a reference section summarizing the main takeaways as well as providing some details on technical aspects that are not discussed in the main part of the chapter.

Big Idea

Data is *distributional* because of inherent randomness in the physical world. We can visualize data distributions in our computers and use both theory and simulation to compute interesting quantities using these distributions.

1 Coding and Plotting

As mentioned above, we believe that a modern quantitative researcher knows not just how to run programs and canned scripts, but more generically how to use and generate computer code as a tool that supplements all aspects of quantitative experimentation. The Python tutorial linked here contains the minimum coding details needed to complete the assignments and worksheets contained in this course. While a completely novice coder may struggle to quickly complete these tasks, it is our goal that there is sufficient support in these resources that anyone can follow along.

In particular, to complete the worksheets and "Try It Yourself" tasks, you should complete the tutorial except for the sections on dictionaries and classes. The sections on random numbers, plotting, and loops will be especially useful in this module.

2 The Atom of Probability

While we hope to convince you that the methods and frameworks presented in this text will at least be useful, it is worth spending some time arguing that a probabilistic view of the world is actually the *correct* view in that it most accurately describes and incorporates all observed phenomena. This may sound philosophical, and it is, but it is also practical. If you are a generic empirical observer this practicality is two-fold.

First, let's say that you have an experimental measurement that you believe is *deterministic* (whatever that may mean), this measurement is inevitably corrupted by measurement error, where we refer to error intuitively as a discrepancy between what we observe and the true value of the phenomenon we are observing. This error might be due to the measurer, the precision of the measuring instrument, or some other unknown effects. Secondly, and more directly, any physicist or chemist can attest that *all* phenomena are stochastic (random), either due to thermal or quantum mechanical effects. The relevance of this stochasticity depends on the phenomenon, but generally as the scale of the phenomenon becomes smaller (in time, size, number of samples, etc.), the more important this physical randomness becomes. These physical considerations place fundamental limits of reproducibility in the phenomena and measuring it.

Given the inherent noisiness or error in all measurements, it should seem necessary to have methods to account for it. That is, given that I know that there is some element of randomness in my observations, how can I make inferences or conclusions from any measurement? Probability theory was created precisely to provide a framework in which to

understand the inevitable randomness in any real measurement.

2.1 Basic Probability Theory

While we won't spend much time worrying about the theory, it is useful to spell out a few basics. In particular, the base premise of probability theory is that all events are **random variables** that take on different values with different **likelihoods**. That is, the roll of a single die is a random event where the value of 1 occurs with likelihood 1/6, the value 2 occurs with likelihood 1/6, etc. Mathematically, we use the notation of a capital letter, X for example, to indicate a random variable, and its lower case, x, to indicate a specific outcome of that random event. So in the example, X indicates the roll of a die, and $x = 1$ is a specific outcome. To make statements about likelihood, we then can write

$$P(X = 1) = 1/6,$$

which is read, "the probability that the outcome of a dice roll is 1 is one out of six ($1/6 \approx 16.667\%$)." The $\mathbf{P}()$ indicates that we're making a statement about the likelihood or *probability* of whatever is being written in the parentheses. We we write this generically as $P(X = x)$ and we will often use the abbreviated form $P(X)$ to mean the same thing.

A modern statistics textbook would then go on to elaborate on some rules of probability, but these tend to be more details than essential to an understanding of the topic, so we'll only discuss them at the end of the chapter. However, it is worth thinking briefly about the concept of **dependent** and **independent** random events. These definitions are mostly intuitive; two random events are independent if their outcomes are independent of one another, they are dependent if one of them depends on the other. For example, if I have two coins and I flip one, the likelihood that the other one will be heads doesn't depend on the outcome of the first flip. On the other hand, if I have a deck of cards and I draw two cards, keeping track of the order in which they were drawn, the likelihood that the second card is the ace of spades *depends* on whether I drew the ace of spades as my first card.

Mathematically, independent random events have the property that their likelihoods *multiply*:

$$P(X \text{ AND } Y) = P(X) \times P(Y) \tag{1}$$

To be explicit, $P(X \text{ AND } Y)$ is the **joint probability distribution** of X and Y. Note that this is the abbreviated notation, and what we mean when we write this is that we are interested in the likelihood that X takes on some outcome, x, *and* that Y takes on a specific

outcome, y, so that we might also write $P(X = x \text{ AND } Y = y)$. In this way, can unpack Equation 1 in words: if X is the roll of a die and Y is the flip of a coin, then the likelihood of rolling a 1 and flipping a heads should just be

$$P(X = 1 \text{ AND } Y = H) = \frac{1}{6} \times \frac{1}{2} = \frac{1}{12}.$$

Hopefully you can see that this holds in general: if two things don't depend on each other, then the likelihood of both of them can be found by multiplying their individual likelihoods.

To see why this is a useful consideration to make, we introduce the concept of a **conditional probability**, which we denote $P(X \,|\, Y)$. In words, we read this as "the probability of random variable X taking on a specific outcome *given* a specific outcome for Y." So in the example of drawing cards, if C_1 is the first card I draw and C_2 is the second, then the probability of $C_2 =$ ace of spades can be notated

$$P(C_2 = \text{ace of spades} \,|\, C_1) = \begin{cases} 0 & \text{if } C_1 = \text{ace of spades} \\ \frac{1}{51} & \text{if } C_1 \neq \text{ace of spades.} \end{cases}$$

So *given* the first card that I drew from the deck, the probability of the outcome of the second card changes. This is completely different from two independent random events, where the probability of flipping heads, for example, doesn't depend on whether I just flipped heads or tails (or rolled a die, or measured the temperature, or went for a walk, etc.).

If we then want to know the *joint probability* of getting two specific cards C_1 and C_2, then we can write the joint distribution more generally as

$$P(C_2, C_1) = P(C_2 \,|\, C_1) \times P(C_1), \tag{2}$$

where we've now used the further abbreviation of $P(C_2, C_1)$ to refer to the joint distribution of C_1 and C_2. This equation is the general definition of a joint probability distribution, regardless of whether C_1 and C_2 are independent. If C_1 and C_2 were independent, then Equations 1 and 2 would be equivalent because for independent random variables

$$P(C_2 \,|\, C_1) = P(C_2),$$

as we'd expect. That is, if they are independent, then the probability of C_2 does not depend on C_1. (The above equation is an alternate definition of independence.)

This might all sound somewhat tautological, but it can be somewhat subtle and it will

continue to come up as you progress through the course. Most importantly, we want to get you used to the notation and how you should read these mathematical statements to yourself. In particular, you should make sure you understand the difference between two events being independent or not. Two measurements being independent does not just mean that they were taken with different instruments or by different people. If I measure the temperature at noon and you measure the humidity 4 hours later in the same location, those quantities are not independent; the likelihood of 60% humidity definitely depends on whether it was 0°F or 100°F at noon. (If this weren't the case, we'd have no ability to predict the weather at all!)

As a result, you should consider independence to be the exception rather than the rule, and we prefer our quantities to be independent for mathematical and theoretical reasons. Generally, we assume that our data are composed of independent samples (of one or more quantities that may or may not be independent of each other), but for many of the methods presented in these notes this is not essential. We will attempt to be clear when independence is an implicit assumption to any method.

2.2 Flipping Coins

Now that we have gotten some formality out of the way, let's get to our first big concept: flipping coins. The toss of a coin is often called the "hydrogen atom" of probability and statistics because just as a hydrogen atom is the building block for theory in physics and chemistry, the coin toss is our elemental unit.

To be precise, let's consider a coin toss to be a random event, X, with two possible outcomes: heads (H) and tails (T). Let's say that the likelihood of getting a heads is p so that

$$P(X = H) = p \qquad \text{and} \qquad P(X = T) = 1 - p.$$

Why must $P(X = T) = 1 - p$? This is a (hopefully intuitive) rule that the probability that *any outcome* will result from a random event is 100%. If I flip a coin, I'll get heads or tails; something has to happen. So if the likelihood of heads is $p = 4/10$, then since tails is the only other option, $P(X = T) = 6/10$. This is a useful property that we'll make use of frequently.

2.2.1 Why are Coin Tosses Random?

Now, while you've probably been told your whole life that a coin toss has a random outcome, you may be wondering how exactly (physically) this can be. To better understand this, it is instructive to consider a paper by Joe Keller, one of the giants of applied mathematics in the 20$^{\text{th}}$ century. In this paper, Keller treats the coin as a regular Newtonian object, subject to the force of gravity. He notes that tossing a coin is a relatively simple mechanics problem: it has some initial conditions (at $t = 0$, the coin is given some rotational and translational inertia), and some governing equations of motion (Newton's laws). Given these initial conditions and equations of motion, you can relatively simply have your computer calculate the consequences: starting with u angular velocity and ω translational velocity, the coin will come up heads or tails. If you change the initial conditions, the outcome will change correspondingly. (In his analysis, he ignores air resistance, bouncing, and the option to land on the coin's side, which actually makes his point more salient.) So then how is it that this deterministic process is our base model for a random process?

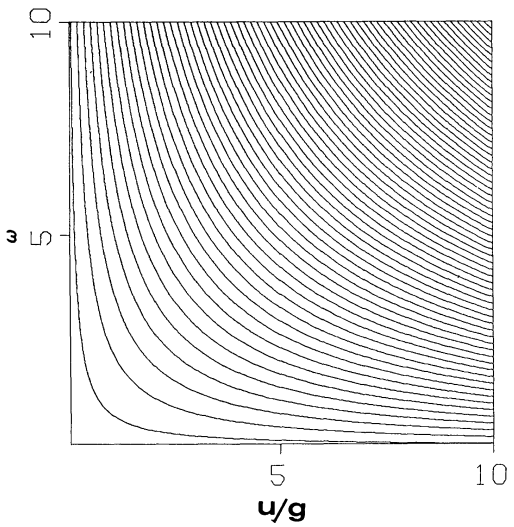


FIG. 2. The curves which separate the sets H and T, the preimages of heads and tails in the u, ω plane of initial conditions, are shown for various values of n. These are based upon (11), with the abscissa being u/g. The lowest strip, adjacent to the axes, belongs to H, the next to T, and so on alternately.

Figure 1: Figure 2 from *The Probability of Heads* by Keller (1986).

What Keller shows is that while a coin flip is an entirely deterministic process, the outcome is uncertain due to an inability of any human (or machine!) to prescribe the initial conditions in such a way as to reliably get a heads or tails. Figure 1 shows the outcome of a coin toss for an initial angular velocity, ω, and vertical velocity, u. The black lines denote the edges of regions of this parameter space that give rise to heads or tails. It's not

surprising that these bands alternate between heads and tails, but what is surprising is that the bands alternate increasingly rapidly as ω and u increase. The bands become increasingly close together so that even the tiniest change in the spin or velocity of the coin will change it from a heads to a tails. This is an example of the concept of *chaos*, which is when the outcome of a system is enormously sensitive to initial conditions. (This is also known as the Butterfly Effect.) Bringing up this analysis serves to underline that even the most simple systems may need to be considered probabilistically, even if for no reason other than that the mechanistic description is less useful.

2.3 Flipping Several Coins

Accepting then that a coin is a random process, it may not yet be clear how we can learn much from flipping one coin at a time. In fact, one coin is not terribly exciting, but if we flip, say, N coins, things become much more interesting. Most obviously, flipping N coins has many more possible outcomes, from N heads to N tails and everything in between. If we want to write down the probability of observing say, 8 heads out of 20 coin tosses, we can take the likelihood of getting 8 heads and 12 tails: $p^8(1-p)^{12}$. (Why does this formula make sense?) However, there are many ways to get 8 H and 12 T, the first 8 flips can all be heads and the last 12 can all be tails, or the first 7 can be heads and the next 12 tails and the last a heads again, etc. To account for this we use the combinatorial factor $\binom{20}{8} = \frac{20!}{8!12!}$, where the "!" is the **factorial** operator ($5! = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$, for example).

Writing the above more generally, we get

$$P(k \text{ heads in } N \text{ flips of a coin}) = P(k \mid N, p) = \frac{N!}{k!\,(N-k)!} p^k (1-p)^{N-k}, \tag{3}$$

which you may recognize as the **binomial distribution**. We have not yet discussed distributions, but they are the central concept to this entire text. Most broadly, a probability distribution or **Probability Density Function** (PDF) can be understood as a description of how specific outcomes of a random variable are *distributed* (how the *density* of probability is spread out). So in the case of a single coin with $P(X = H) = p$, the PDF is written

$$P(X = x) = \begin{cases} p & x = H \\ 1-p & x = T \end{cases}$$

to explicitly state the coin flip distribution. The binomial distribution then describes how the different number of heads in N coin flips are distributed.

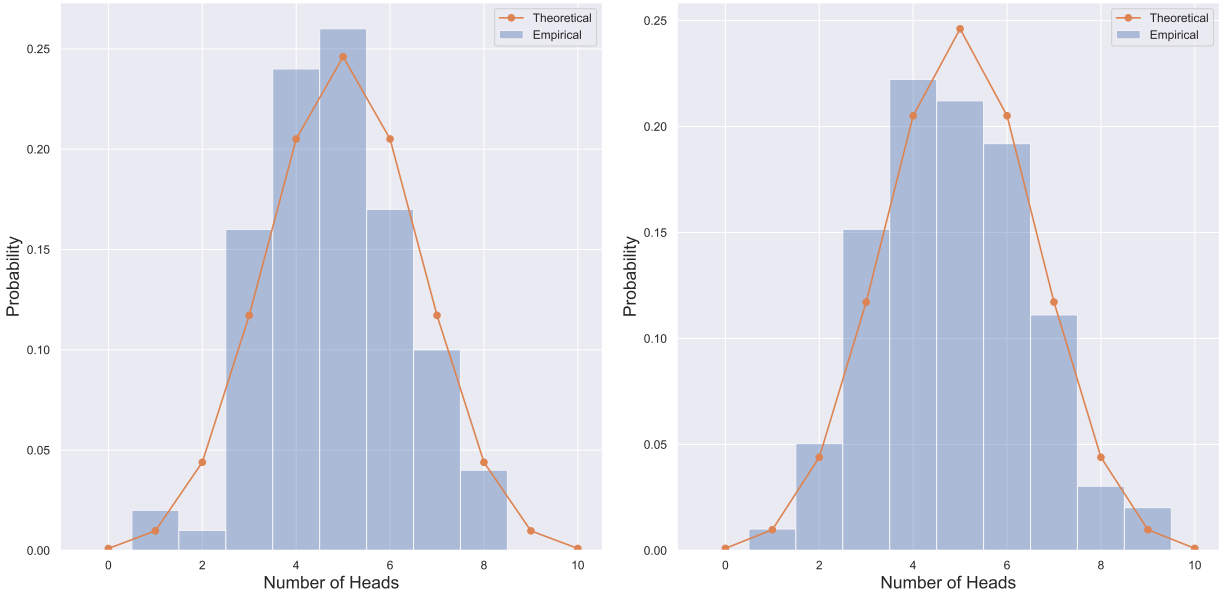


Figure 2: Visualization of two empirical distributions generated from flipping 10 fair coins 100 times. The theoretical binomial distribution is plotted over each in orange.

Probability distributions, loosely speaking, come in two flavors: empirical and theoretical. Empirical PDFs (ePDFs) are of the kind that you'll generate in the worksheets and problem sets; generated from a finite amount of data and proportional to a frequency distribution. Theoretical probability distributions instead (usually) have a closed mathematical formula, and can be thought to represent the limiting empirical distribution in the case where you have infinite data. An example of each can be found in Figure 2 where two different instances of an empirical distribution are shown as the blue bars and a theoretical PDF is shown by the orange line in each panel. As an analogy, the difference between empirical and theoretical probability distributions is similar to the difference between the equation of a circle, $x^2 + y^2 = 1$, and any real world drawing of a circle. The real world drawing approximates the circle, and there are ways to improve this approximation, but the equation is always a Platonic ideal that cannot be achieved due to the finite nature of the physical universe. Similarly, repeated constructions of an empirical probability distribution will result in slightly different approximations to the ideal, as can be seen in Figure 2.

While this may sound somewhat philosophical, it is crucial to understand that any real world measurement is an empirically acquired outcome of some random process, and therefore will change upon repeated observation. Hopefully you can see that while we would ideally like to make many observations, so as to put ourselves closer to the regime of the theoretical distribution, this is obviously not always the case. In fact, it is relatively common to find

ourselves in the data-poor limit of an experiment, so it's very important to be especially cognizant of the fact that we have empirical observations. As an example, if you toss a fair coin 10 times, it is very possible to get 8 heads (this should happen $\sim 5\%$ of the time according to Equation 3!). If you were to then assess that the coin is biased, you would be wrong! The goal of of probability and statistics is to help you separate what is signal from what is noise by allowing you to assess the likelihoods of different outcomes.

2.4 Features of Distributions

One of the most fundamental insights in all of science is that while the outcome of a single experiment, say a single coin toss, is uncertain, there is a regularity to the outcome of many experiments. You will see this play out in the worksheets and assignments, but you should also recognize this from your own life; if you were to measure a table's length with a ruler and get 1.8 meters, and then you measured it again, you would not get 12 meters. You may however get 1.81 or 1.79 meters depending on how you angled your ruler. If you made many measurements of the table's length, you would probably get a very narrow distribution of table lengths centered at 1.8 meters, and you would be very justified in reporting that the table is 1.8 meters long.

This may seem somewhat trivial, but the point is this: we've elaborated that measurements are random events where the likelihood of any outcome has some (unknown) distribution, and we've noted that repeated observations allow us to get closer to a theoretical distribution, so when we are taking measurements of some quantity, we're not trying to get a better list of numbers, we're trying to *discover the underlying distribution* that governs how a quantity is observed. That is, we want to know about a quantity's probability distribution in order to characterize it. Furthermore, we can assert that all quantities are actually distributions from which we draw observations.

While this may seem somewhat abstract - and it is - what it means to you as a quantitative thinker is that we should be very concerned with the *shape* of our observations' probability distribution. It may not be clear why we can expect our observations to have any regularity, and we'll get to that shortly, but for now, let's discuss what sort of features of our distributions are interesting.

2.4.1 Moments of Distributions

Perhaps most obviously, we might be interested in the most frequent outcome of a random event (this is known as the **mode** of the distribution). For example, if you were given a coin

with an unknown bias, you can imagine conducting many experiments of many coin flips and looking at what the most frequent result was to determine whether the coin is fair or not. That is, we can potentially (hopefully) infer the fairness of the coin by observing properties of its probability distribution.

However, as we'll show in assignments and in later sections, looking at peaks of distributions is not the only useful feature. For example, consider Figure 2, in which the distributions for the number of heads in 10 coin flips are shown after performing the experiment (10 coin flips) 100 times. According to the left panel, the mode is 5 heads (in 10 flips), and according to the right panel, the mode is 4 flips. If we had only this information, it would be very hard to know whether the same coin was used to generate the distributions (it was!).

In fact, there are some general mathematical ways to quantify different features of a distribution. In particular, we can talk about the **mean** or **expectation** of a distribution, which can be calculated either from a theoretical probability distribution

$$E[X] = \int_{x \in X} xP(X = x)dx,$$

or from an empirical distribution

$$E[X] = \sum_{x \in X} xP(X = x), \tag{4}$$

where Σ is the summation operator, so we want to add up x times $P(X = x)$ for all possible values of x. So for example, if we consider X to be the roll of a fair die, then x can be either $1, 2, 3, 4, 5$, or 6 and the full sum can be written

$$\begin{aligned} E[X] &= (1) \cdot P(X = 1) + (2) \cdot P(X = 2) + (3) \cdot P(X = 3) \\ &\quad + (4) \cdot P(X = 4) + (5) \cdot P(X = 5) + (6) \cdot P(X = 6) \\ &= 1 \cdot \frac{1}{6} + 2 \cdot \frac{1}{6} + 3 \cdot \frac{1}{6} + 4 \cdot \frac{1}{6} + 5 \cdot \frac{1}{6} + 6 \cdot \frac{1}{6} = 3.5, \end{aligned}$$

and we expect the value of the die to yield 3.5 on average. (Make sure this makes sense to you!) Note that we talk about expected values of *random variables*, X, not particular outcomes ($x = 1$, for example); the outcome of a random variable is uncertain, but if the die came up 1, then that's our value.

If we think of the mean as a measurement of a distribution's *center*, how can we

describe its width? One measure of "width" is the **variance**, which can be calculated

$$Var[X] = E[(X-\mu)^2] = E[X^2] - E[X]^2, \tag{5}$$

where $\mu = E[X]$ and $E[X^2] = \sum_{x \in X} x^2 P(X = x)$. Using this formula, we find that the variance of a single die roll is

$$Var[X] = 1 \cdot \frac{1}{6} + 4 \cdot \frac{1}{6} + 9 \cdot \frac{1}{6} + 16 \cdot \frac{1}{6} + 25 \cdot \frac{1}{6} + 36 \cdot \frac{1}{6} - (3.5)^2 = \frac{91}{6} - (3.5)^2 = 2.9167.$$

The **standard deviation** of a set of observations is another common measure of spread and is defined as the square root of the variance ($\sigma = \sqrt{Var[X]}$).

You can imagine that there are a whole set of measurements $E[(X-\mu)^n]$, and indeed, these are called the **moments** of a distribution ($E[(X-\mu)^n]$ is the n^{th} moment). In fact, as n increases, you learn about the shape of the distribution further and further from the mean. Specifically, the odd values of n give information about how asymmetric the distribution is and the even values about how symmetric the distribution is. A certain version of $n = 3$ is called the **skewness** and $n = 4$ is known as the **kurtosis** of the distribution.

Try It Yourself

To put this theory into practice, take a stab at Worksheet 1.1.

Once you have completed this worksheet, you should have enough practice to start to work on Assignment 1!

2.4.2 Cumulative Distribution Functions

One important note to make here is that when we have empirical distributions, we need to be careful about how we define $P(X)$. Say for example, you were measuring the heights of everyone at Northwestern, but rather than keeping track of thousands of heights, you were just counting how many people's heights fell into a set of bins, e.g. 120cm-130cm, 130cm-140cm, etc. (Would this be a distribution??) As you'll explore in worksheets and assignments, your choices as to bin size and number will affect any calculations of moments that you make later on.

Try It Yourself

Change the bin sizes in your histograms and distributions in Worksheet 1.1. That is, instead of showing how many times you observed 5 heads, 6 heads, 7 heads, etc. Show how many times you observed 0, 1, or 2 heads, the number of times you observed 3, 4, and 5 heads, etc. How do any calculations based on the PDF change? Try and modify any curves generated using theory to use the same bins.

While this is not catastrophic, it's also not desirable, so it is worth noting that we can also characterize the shape of a distribution using **percentiles**. A percentile is defined as the value of an observation such that a certain percent of all the observations are less than or equal to that value. More formally:

$$Q_X(p) = \min_{x \in X} \left\{ x \,\middle|\, \sum_{\chi \leq x} P(X = \chi) \geq p \right\}$$

To calculate percentiles empirically, we will want to generate a **cumulative distribution function**, or CDF (we'll sometimes denote an empirical CDF as eCDF to distinguish from theoretically derived CDFs):

$$F[x] = \int_{-\infty}^{x} P(X = x)dx.$$

In words, the CDF describes the likelihood of getting an outcome less than x, and as such is also sometimes written $P(X \leq x)$. We can do this empirically using the following Python code:

```
import collections

## Counts the number of times each value in data is seen
counts = collections.Counter(data)

## Gets unique values of data and sorts
vals = np.msort(list(counts.keys()))

## List comprehension to get ordered counts for each value
## cumsum then cumulatively sums these counts
CDF = np.cumsum(np.array([counts[ii] for ii in vals]))

## Normalize so that the PDF adds to 1.
CDF = CDF / CDF[-1]
```

Using this code, we can generate CDFs for the data shown in Figure 2, which are shown in Figure 3. Then, using the CDFs, we can easily read off percentiles by finding the desired percentile on the y-axis and finding the corresponding x-coordinate on the CDF. For example, in Figure 3, we can see that the 20th percentile of both distributions is approximately $k = 3$ heads.

Particularly interesting quantities are the **median**, which is the 50th percentile, and the **Inter-Quartile Range** (IQR), which is the distance between the 25th and 75th percentiles (the first and third **quartiles**). In both the examples in Figure 3, the median and IQR are 5 and 2, respectively. The median and IQR are often more useful for characterizing a distribution than the mean and variance because they are *robust to outliers* in your observations.

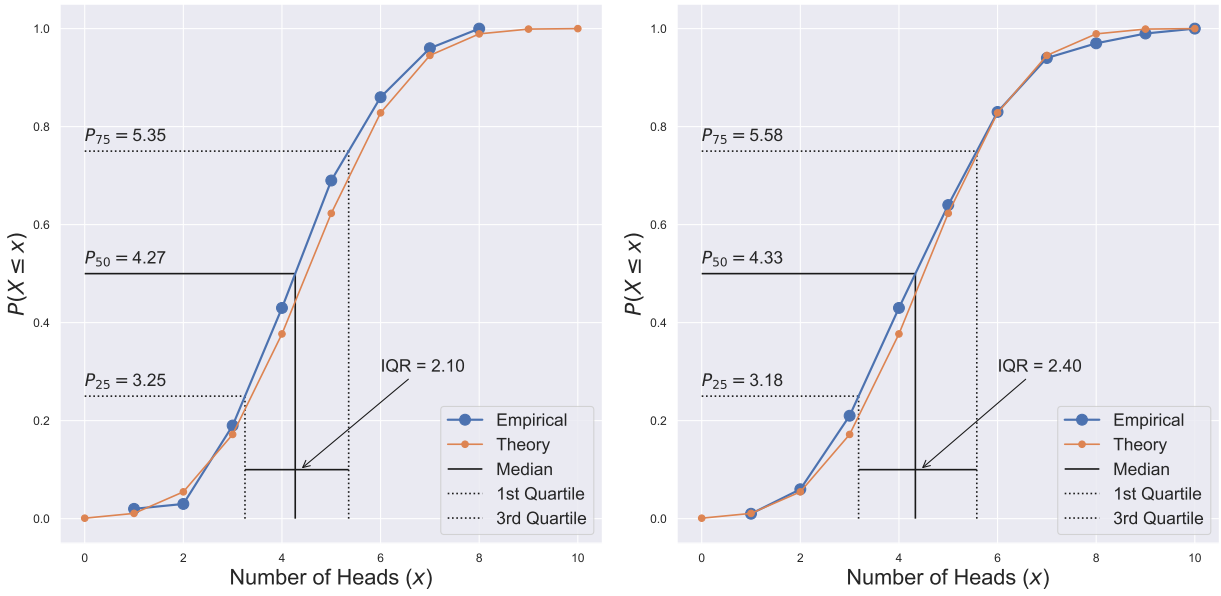


Figure 3: Empirical and theoretical CDFs for the number of heads in 100 experiments of 10 coin tosses. Note that in neither of the 100 experiments was 0 heads observed, so neither empirical CDF extends to $x = 0$. Similarly, in the left panel's 100 experiments, no observations of 9 or 10 heads were made either. The median, 25th, and 75th percentiles are also shown, and the IQR is indicated. FIX FIGURE TO NOT INTERPOLATE!!!

It's worth mentioning that while the binomial distribution is a useful and intuitive distribution to know, it is certainly not the only theoretical distribution worth knowing about. In particular, **Normal** or **Gaussian** distributions and **Poisson** distributions are often useful in quantitative analyses. However, their functional forms are not essential for

the understanding of the main part of this chapter, so we will leave them to the end.

Try It Yourself

Use `numpy.random.randn` to generate 2 lists of 10 normally distributed random numbers. Add the value 100 to one of the lists. Compare the mean and median of the 2 lists. Which is the better descriptor of the "average" value of the list, the mean or median?

2.5 The Central Limit Theorem

In the previous sections, we've mentioned that theoretical distributions are the infinite-data limit of observational (sampling) distributions, but we have not provided any rigorous reason why our approximations get better as we make more observations. We will not do so here except to highlight the famous Central Limit Theorem (CLT), whose many proofs provide the basis for our assertion. We will not prove the CLT here, instead we find it most useful just to highlight its main result and assumptions.

Specifically, the CLT says that in an extraordinary number of cases, the ratio of the standard deviation (σ) to the mean (μ) of a set of observations decreases as the number of observations, N is increased. Moreover, the ratio decreases in a specific fashion: as the square root of N, as shown below.

$$\frac{\sigma}{\mu} \sim \frac{1}{\sqrt{N}} \tag{6}$$

More intuitively, if we think of the mean as a measure of the "center" of our distribution, and the standard deviation as the error in estimating the true mean, then our ability to estimate the first moment of our distribution increases as the number of data points is increased.

Now you might see how this could be useful. If we're concerned about separating signal (μ) from noise (σ), then we can use the CLT to determine exactly how many observations we need to estimate the mean to a certain accuracy. However, the CLT requires a few assumptions about the data in order to make this statement:

1. all measurements are independent of one another, and
2. all observations must be generated from the same underlying process.

In terms of a coin toss, as long as each coin being tossed has the same value for $P(X = H) = p$, then tossing increasing numbers of coins improves our empirical estimation of p. That is, the CLT tells you that if you continue flipping a coin enough times, you can be increasingly

confident calculating p. Hopefully this is reassuring and intuitive: adding more data should improve our inferences and estimates.

Try It Yourself

Consider the experiment from Worksheet 1.1 where you are flipping N coins repeatedly. If I want the standard deviation in the number of heads to be *on average* 10% of the mean, how many coins do I have to flip? Show this with code if you can (or if you're having trouble with the theory)!

The flipside of this is that when we have very few measurements of a phenomenon, then our ability to estimate a given feature of the distribution, say the mean, will be error-prone. The CLT tells us that the error in these estimates will often scale with the inverse square root, as shown in Equation 6, but that is not much comfort if we only have 2 or 3 measurements. This is worth keeping in mind whenever you are working with data, even if it is "big"; there is a difference between having a few measurements of many different quantities and many measurements of just one quantity.

If this were all the CLT gave us, it would be a useful result, but the CLT also can be stated (in combination with the Law of Large Numbers) to say that if you have enough measurements (if N is big) of an independently and identically distributed (i.i.d.) random variable, then the distribution of the mean of those measurements can be characterized completely with only the mean and standard deviation (the first two moments) of the random variable. For those with some previous experience in these topics, the CLT says that the *sum* of i.i.d. random variables becomes a Gaussian distribution as the number of observations, N, increases. Even more specifically, if the i.i.d. random variables have mean μ and standard deviation σ, then the distribution of the mean has mean μ and standard deviation $\sigma/\sqrt{N}$. This may seem like an odd thing to focus on but it's actually very useful as we now know how *any sum* of i.i.d. variables is distributed. While in most of this course we will stay away from making assumptions as to the form of any distribution, it is quite remarkable that all sums of random variables are distributed similarly.

The upshot of the CLT is this: when the assumptions are met, it is a very useful result! However, in many situations we do not have many i.i.d. random variables, so it is not prudent to preoccupy oneself only with analyses that rely on the CLT to work. Instead, we hope to build up some methods that will apply even when the assumptions of the CLT are not strictly valid.

Try It Yourself

Try your hand at Worksheet 1.2. You should also now be able to attack all of Assignment 1 except for Problem 3.f.

3 The Atom of Bayesian Probability

Throughout this text, our goal is to remove obstructing details and formulas from your path towards becoming a quantitative worker. While some details are inevitably necessary in any field, what we have been emphasizing are the most important and most useful parts of theory for actually learning things from data. If you have previously taken a statistics course, it may come as a surprise to you then that we think it's worthwhile to talk about Bayes' Theorem. Our motivation here is partly philosophical, but also (as always) practical.

First, Bayes' Theorem can be written explicitly:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}. \tag{7}$$

As noted earlier, the expression $P(A|B)$ is read "the probability of random event A *given* random event B" and is called a **conditional probability**. This name is due to the fact that we are thinking about the likelihoods of outcomes of random event A *conditioned on* (depending on) the outcome of random event B. In this way, Bayes' Theorem is simply a formula relating the conditional probabilities of two random events.

Try It Yourself

Using Equation 2 in two symmetric ways, see if you can derive Bayes' Theorem yourself.

To make it clear why Bayes' Theorem might be useful, let's consider random event B to be the act of observing data and random event A to be some interesting feature of the data distribution, let's say the mean. Then as a data researcher, we might be concerned with knowing that interesting feature, and if we're being careful, we might be very concerned with *how well* we can know that feature *given that we collected certain data*. That is, we are almost always concerned with knowing $P(\text{FEATURE}|\text{DATA})$, the likelihood that our feature has a specific value given our specific data. This quantity is known as the **posterior** distribution for the feature A.

$$P(FEATURE \,|\, DATA) = \frac{P(DATA \,|\, FEATURE)P(FEATURE)}{P(DATA)} \tag{8}$$

The reason then that Bayes' Theorem is so appealing is that it gives us an explicit structure in which to calculate this conditional likelihood. Specifically, if we can construct the parts of the right-hand side of Equation 7, we can plug them into Bayes' Theorem and we have our answer. You may be concerned however as to how we might know the parts of the right-hand side better than the left, and this is a main criticism of Bayesian statistics, however Bayesian practitioners prefer to keep their assumptions front and center (as a main part of any calculation), than not. But before we get into the philosophical side, what even is on the right-hand side of this equation?

Keeping the analogy that A =FEATURE and B =DATA, then $P(DATA|FEATURE)$ is the opposite conditional statement to the one we're interested in, and is called the **likelihood function**. That is, it is the conditional likelihood of observing our data given some feature of the sampling distribution. In many cases (we'll get into this), we have a **model** for how our data are generated, and this model may depend on some **parameters**, which may be features of a distribution. In any case, when we have such a model, Bayes' Theorem accounts for it with this term.

The term $P(A)$ is known as the **prior** and it represents our *prior* knowledge about how we think that the feature A is distributed *before* we collect any data. You may be inclined to assert that we don't have any information about A, otherwise we wouldn't be doing experiments to measure it, but this isn't really true - we often do have some guesses as to the shape or size of A. For example, if I hand you a coin and tell you it's from the U.S. Mint, it would not be entirely reasonable of you to state that you have absolutely no notion as to how often the coin will land on heads. Based on personal experience alone, it would be more prudent to start from the assumption that the coin is probably fair and wait for new evidence to convince you otherwise. It's worth mentioning that even if you want to insist that you don't know anything, you can often plug in what is called a **non-informative** prior, for example, a uniform distribution for p from 0 to 1 in the case of the coin toss. Also, regardless of your choice of prior, given enough evidence Bayes' Theorem will always converge to the same posterior distribution, so this seemingly arbitrary choice doesn't often end up having that large of an impact. (We'll explore this more concretely in a worksheet later.)

Let's now work a specific example to see how this works.

3.1 Example: Bayesian Coin Tossing

While we'll talk about parameter estimation more in the next section, let's consider the scenario where you have been given a coin and you want to assess whether it is fair. We'll use Bayes' Theorem in a slightly looser formulation:

$$P(p \,|\, k \text{ Heads in } N \text{ tosses}) \propto P(k \text{ Heads in } N \text{ tosses} \,|\, p) \cdot P(p)$$
$$P(p \,|\, k, N) \propto P_{Binom}(k \,|\, N, p) \cdot P(p),$$

where we omit the denominator, $P(k \text{ Heads in } N \text{ tosses})$, because we know that the left-hand side is a probability distribution and thus must sum to 1. Let's then say that $P(p)$, our prior for the distribution of the heads probability, p, is a uniform distribution from 0 to 1 (often denoted $\mathcal{U}(0, 1)$). That is, we're asserting that we really don't know anything about coins.

We know from our earlier derivation of the binomial distribution, that if we know p, then we can write the likelihood of observing any number of heads k in N tosses is binomially distributed, so we write

$$P(k \text{ Heads in } N \text{ tosses} \,|\, p) = P(k \,|\, N, p) = \binom{N}{k} p^k (1-p)^{N-k},$$

which is the first factor on the right-hand side of Bayes' Theorem.

Then, before we start flipping the coin, we can visualize our prior for the heads probability in Figure 4. As a counter-point, we also consider a prior in which we strongly believe that the coin will be fair, with much less likelihood that it is biased. This is shown in Figure 5. In this thought experiment, the coin actually is biased to have a heads probability of $p = 0.28$, which is shown in the figures, but of course, we wouldn't know this when we're given the coin.

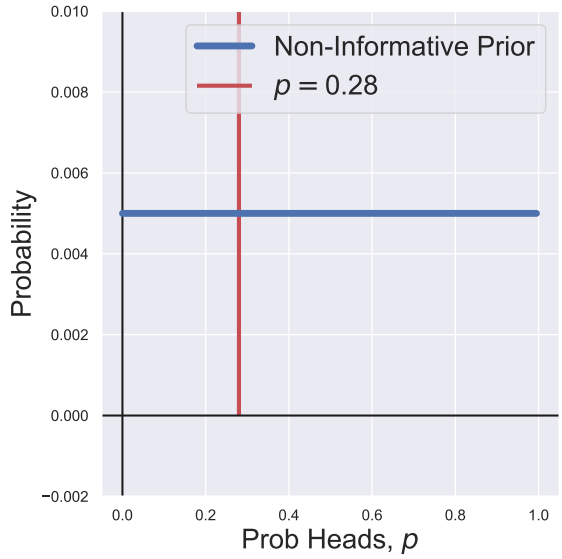


Figure 4: A non-informative prior (uniform distribution) for the heads probability of a coin.

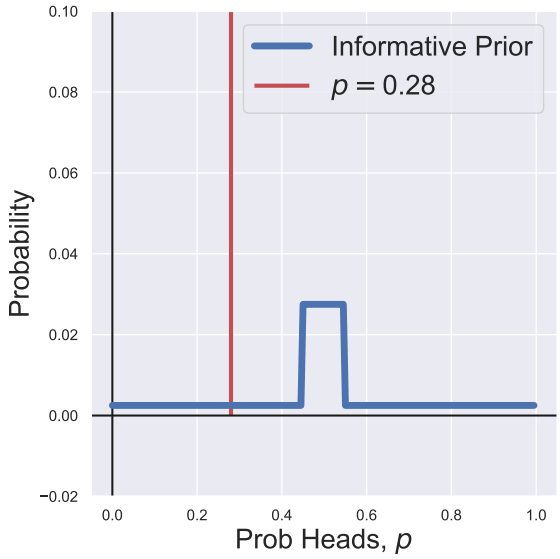


Figure 5: An informative prior (non-uniform distribution) for the heads probability of a coin.

In Figures 6 and 7, we see the result of multiplying $P(0H|N = 1, p)$ and our two priors $P(p)$. That is, we flipped the coin and got a tails, so we shifted our estimates of the likelihood of heads *away* from $p = 1$ and towards $p = 0$, although in the case of our informed prior we're still holding out considerably for a fair coin.

That is, we're showing the function given by

$$f(p) = P_{Binom}(k{=}0 \,|\, N{=}1, p) \times P(p),$$

for two different choices of $P(p)$ (as shown in Figures 4 and 5).

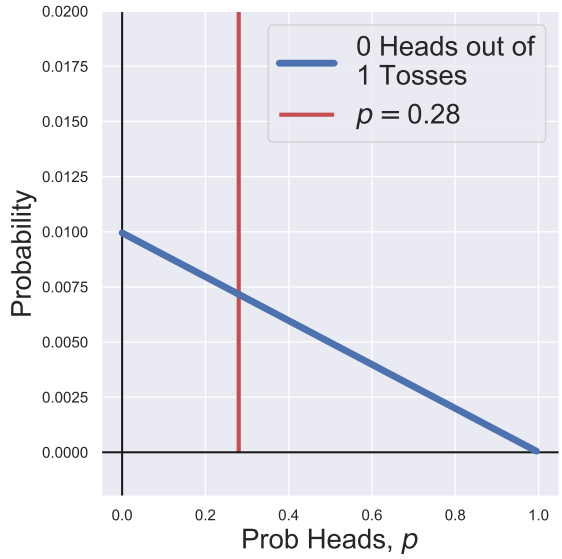


Figure 6: A posterior distribution for the heads probability of a coin after one toss with a non-informative prior.

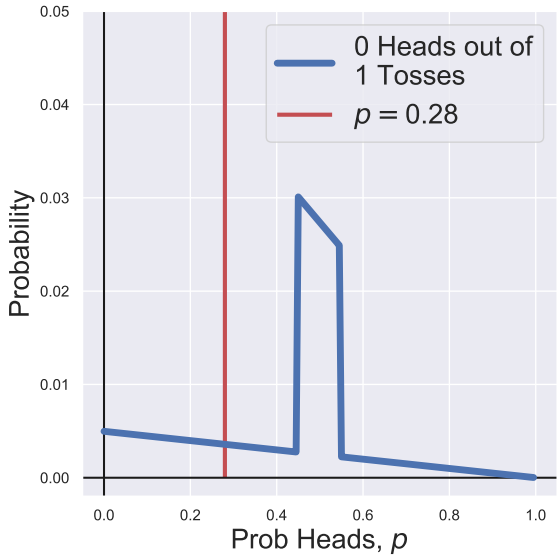


Figure 7: A posterior distribution for the heads probability of a coin after one toss with an informative prior.

In Figures 8 and 9, we've flipped two coins, one of which is a heads, so the posteriors have recentered themselves on $p = 0.5$. Then in Figures 10 and 11, we've skipped ahead to flipping 10 coins, of which 9 were tails! Now we can see that both posteriors are starting to look similar, and the effect of our prior choice is becoming much less signficant.

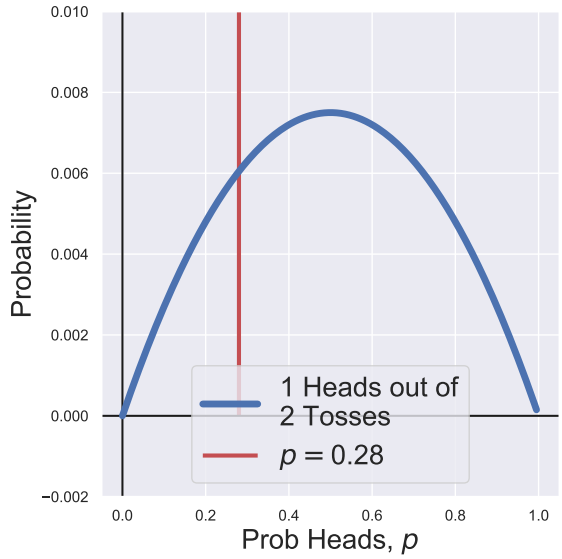


Figure 8: A posterior distribution for the heads probability of a coin after two tosses with a non-informative prior.

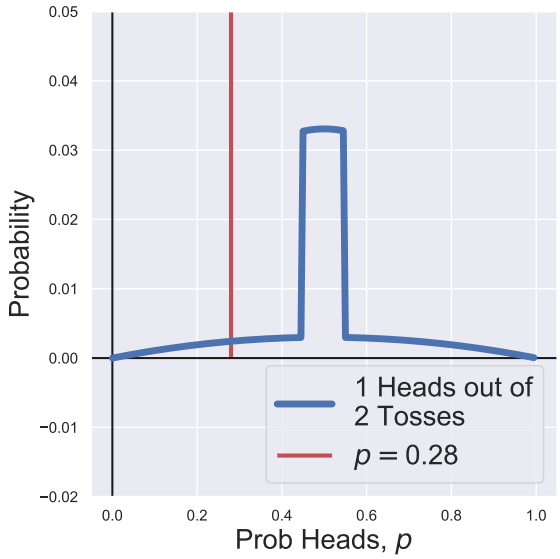


Figure 9: A posterior distribution for the heads probability of a coin after two tosses with an informative prior.

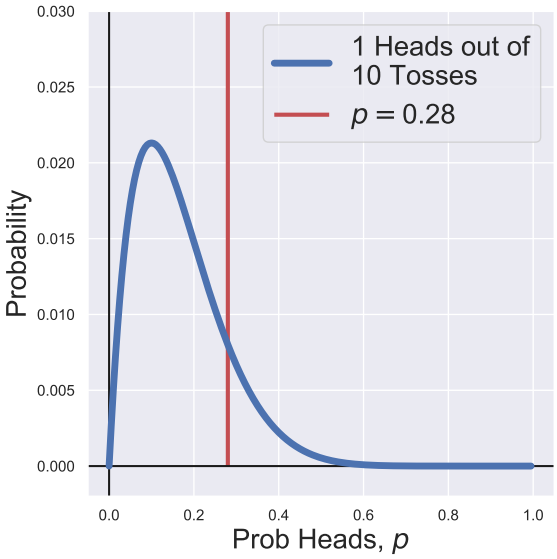


Figure 10: A posterior distribution for the heads probability of a coin after ten tosses with a non-informative prior.

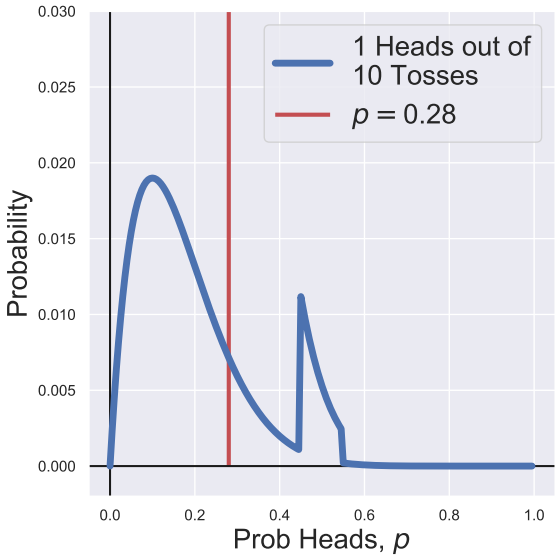


Figure 11: A posterior distribution for the heads probability of a coin after ten tosses with an informative prior.

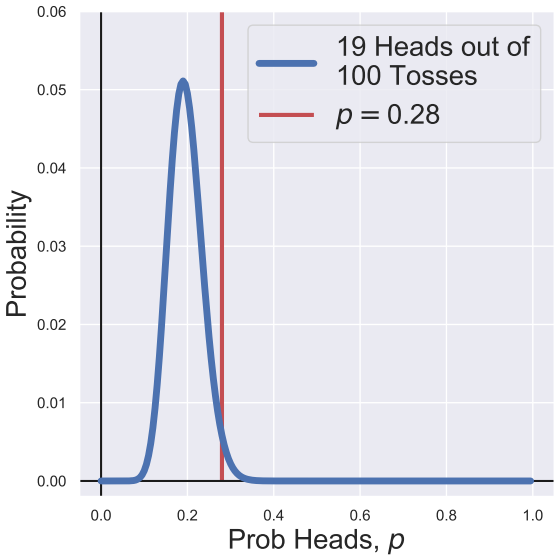


Figure 12: A posterior distribution for the heads probability of a coin after 100 tosses with a non-informative prior.

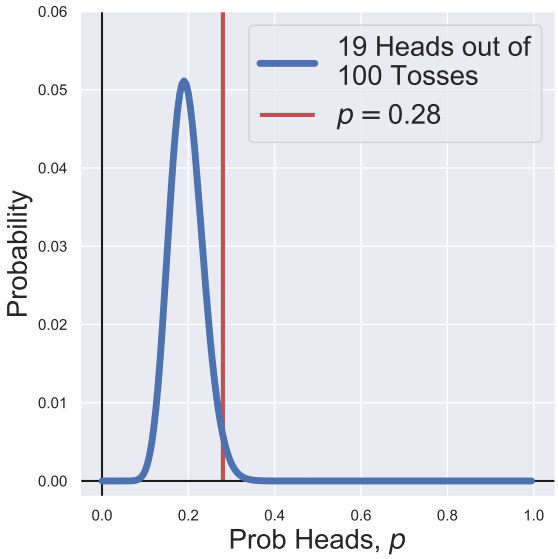


Figure 13: A posterior distribution for the heads probability of a coin after 100 tosses with an informative prior.

Finally, in Figures 12 and 13, we have flipped the coin 100 times, so that any prior expectation for the heads probability is completely wiped out. Both posteriors are (very reasonably) centered on $p = .19$ since we've observed 19/100 coins to be heads, but the true value of $p = 0.28$ still has a non-zero probability associated with it.

More explicitly, we're showing the functions

$$f(p) = P_{Binom}(k{=}1 \,|\, N{=}2, p) \times P(p),$$

$$f(p) = P_{Binom}(k{=}1 \,|\, N{=}10, p) \times P(p),$$

and

$$f(p) = P_{Binom}(k{=}19 \,|\, N{=}100, p) \times P(p).$$

Notice that each of these is a function of our parameter of interest, p, only.

Try It Yourself

You now have the theoretical basis to make an attempt at Worksheet 1.3. Completing the worksheet will likely be necessary before attempting problem (3.f) on the assignment, so make sure you work through the whole worksheet!

3.2 Why Bayes?

At the beginning of the section we described Bayes' Theorem and how it might be conceptually useful, but the example in above also demonstrates a practical aspect of why Bayes' Theorem is good to know: it gives us *distributions* for any features or parameters. We've spent most of this chapter discussing why individual observations should be considered probabilistically, meaning that they are samples from an underlying distribution, so a methodology that naturally is centered on this distributional way of thinking should be appealing.

We won't probe all the nuances and uses of Bayes' Theorem in this chapter, but we will make a point of showing how this relatively simple formula can be extended to most of the applications in this course. For now, we hope that you can start to see how Bayesianism dovetails nicely with a distributional mindset with regards to data; later we'll show that Bayes' Theorem is applicable very generally, even in cases where classical statistics breaks down.

4 Bacterial Chemotaxis

The first biological dataset you will be exploring is related to bacterial chemotaxis. There are beautiful texts (1, 2, 3, and 4), talks (1, 2, and 3), and papers on this topic. As is the theme of this course, we aren't going to go into the details of topics related to biology, physics, or algorithms too deeply. We want to help you explore and extract information from data. We introduce chemotaxis here because it is a particularly clean biological system that produces data that can be analyzed using some of the tools we have already taught you. So, as a reference for the assignments, here is a very quick summary of the overall phenomenon.

Bacteria need to navigate the chemical landscape they inhabit - they need to move towards stuff they like, and away from stuff they don't. This is called chemotaxis: chemically driven motion. In particular, if you place a source of sugar in a solution with bacteria, say a sugar cube, and the bacteria will move towards the sugar. While seemingly simple, over 50 years of genetics, molecular biology, and biophysics has gone into the study of this one phenomenon. Why? Not because its the most crucial biological phenomenon but because it's a system where incredibly precise measurements and experiments can be performed. Fortunately, what we have learned have been incredibly general principals. Indeed, I could (and have) given an entire course in biophysics that solely focused on bacterial chemotaxis.

So, how do bacteria move in response to chemicals? As shown in Figure 14, they have oars, called **flagella**, that propel them through their surroundings. If you watch a single bacterium move you will see that they use their oars in a funny manner resulting in a path that looks like a series of straight lines interjected by abrupt turns, as shown in Figure 14. This particular method of movement is known as making **runs and tumbles**. The way in which the bacteria moves *towards* a sugar crystal is that on the occasions when the runs happen to point *up* the sugar gradient they tend to be longer. When the bacteria is pointed *down* the sugar gradient, the runs are shorter. This simple mechanism ensures that given random orientations after a tumble, the bacteria will have a net movement, averaged over the many runs and tumbles, towards the source of sugar.

So, what produces the runs and tumbles? Indeed the microscopic mechanism for this is rather straightforward. The bacteria has many oars, but instead of them being like the oars for a rowboat, they look like corkscrews. This may seem like a strange choice of oars but there is a very deep reason for this design (if you find this interesting make sure to delve into the books cited above, as well as this beautiful paper). Crucially then, a single bacteria has many of these corkscrew-oars. When the corkscrews spin in one direction (counter clockwise to the direction of motion) then bacteria moves in straight line, making a run. If they rotate

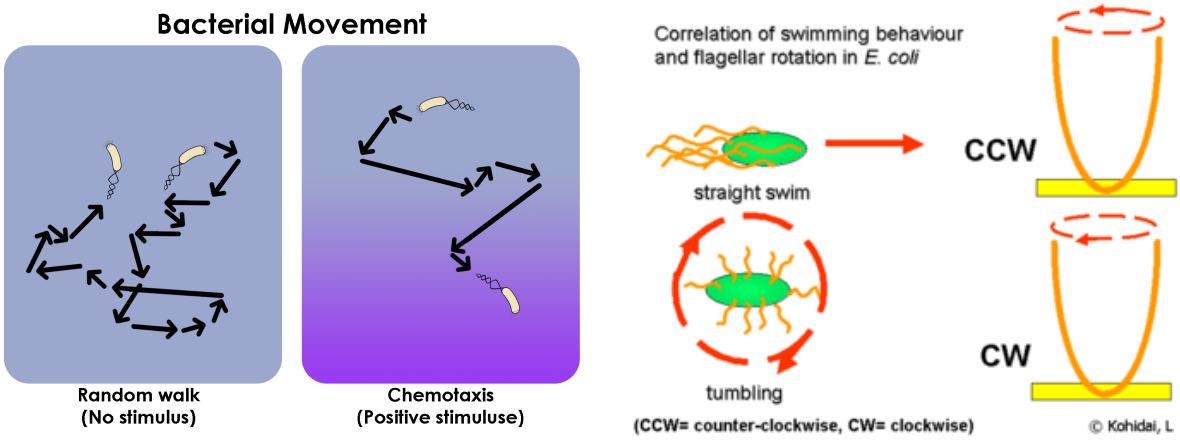


Figure 14: The left two panels illustrate the run-and-tumble trajectories of bacteria in two conditions: the absence of any stimulus (left) and in the presence of a positive stimulus gradient (right). Although it's a cartoon, the idea is that there will be more runs that are longer in the direction of the gradient, compared to the movement when there is no stimulus. The right part of the figure illustrates the physical arrangement of the flagella when they are rotated clockwise and counter-clockwise.

clockwise the bacteria tumbles in place. This is shown in the right-hand side of Figure 14.

At this point then, the question is how a bacteria "knows" that it is moving up the gradient, and thus lengthen its straight line motion? Again, the answer to this involved around 2-3 decades worth of research, but one of the central experiments done to figure this out is what you will analyze as your first data set!

What researchers had difficulty with was in prescribing precise concentration profiles of chemicals and recording how bacteria modulated the frequency with which they did runs vs tumbles. So, to make the observations easier, some researchers chopped off the ends of the long flagella, and rooted the bacateria at the base of a flagellum down onto a slide. Now the bacteria couldn't move translationally, but when the flagella rotated CCW the bacteria would spin CW, and vice versa. Now that the bacteria was spatially fixed, they could then pipette in very precise spatial profiles of chemicals and watch how the bacteria, now stuck in place, would respond. Watching a video of this is somewhat entertaining, but I hope you can appreciate how very simple experimental design choices can be the key to opening up decades worth of insights.

The data that you will be working with is the angular velocity of a bacteria that has been stuck to a slide. Your task in assignment 1 will be in part to verify that you see the bacteria attempting to run and tumble, and to show this phenomenon in a meaningful way.

This data set has been taken from and some of the questions in the assignment have been inspired by William Bialek's book on Biophysics, which I would thoroughly recommend for anyone interested in a more physical viewpoint of biology. Additionally, there is an entire set of notes we have written on chemotaxis that gives a lot more biological and physical background on the topic. Check it out if you are interested!

5 Review

So what was the point of this module? And why should you be reading this? How is this going to help you analyze your data?

A lot of the material here might be something you've seen before. However, getting some definitions and notations down is important. So I hope you have a more precise sense for what $P(X = x)$ is saying. And, for example, how the conditional, marginal, and joint distributions are related to each other. Repeat these concepts in your mind over and over again, with simple examples (tossing two coins, drawing two cards with replacement from a pack, etc.) to really build intuition. I cannot emphasize enough that having these concepts deeply ingrained in your mind will serve you well.

Another really important idea was that even for something as simple as a coin flip you expect a variance in outcomes – a distribution. This makes estimating important features (which we introduced you to, such as the expectation or variance) from empirical data distributions challenging. At a more philosophical level, every statistic (quantity of interest), can only be determined to some finite precision from real data. You saw this in the simple example of coin flipping.

Please reflect on the depth of the central limit theorem within the context of coin flipping. I assure you that this fundamental result is playing a very important role in your data. Intuitively the CLT tells us why getting more data makes the accuracy of any statistic of your distribution better. Additionally, the CLT says that when you have lots of data then often the distribution becomes increasingly "well-behaved" – which means, it becomes increasingly Gaussian. Why do we say that Gaussian distributions are "well-behaved"? Because you can entirely describe a Gaussian distribution using only two numbers – its mean and its variance. And so you need only report two numbers to tell us everything we need to know about the distribution. Generically, the data you generate or want to analyze is actually not Gaussian, or you don't have enough data, which makes characterizing it far more challenging, but we will teach you how to deal with this.

Finally, we briefly introduced a Bayesian point of view to give you some experience with the concepts of the prior, likelihood, and posterior distributions. These 3 can be combined via Bayes' Theorem, which then allows one to exactly state your prior beliefs (the ones you held before you had any data), and update them in the face of new data. What is also beautiful about the Bayesian approach is that any parameter (the probability of heads, even) is now explicitly modeled in a distributional sense. As we'll see in the next module, you won't just get a single number for your estimate but always the whole probability distribution. Of course the price you have to pay is that you almost always need a theoretical formula for the likelihood function. (For coin flips we have the binomial distribution.) You'll continue to encounter the Bayesian perspective as you continue in the course.

5.1 Learning Goals

Below is the curriculum alignment table for this whole module, showing where you can learn about some of the higher level learning goals across the course materials.

Module 1 Curriculum Alignment

Module Components →	**Instruction**							**Assessments**			
Learning Goals ↓	**Notes**	**Video 1.1:** Probability & Definitions	**Video 1.2:** Coin Flipping	**Video 1.3:** Features of Distributions	**Video 1.4:** Central Limit Theorem	**Video 1.5:** Bayes Theorem	Zoom **Discussion Sections**	**Worksheet** 1.1	**Worksheet** 1.2	**Worksheet** 1.3	**Assignment** (Video 1.6 & Video 1.8)
PCS: Calculate summary quantities from data or a distribution	★			★		★	★	★	★		★
PCS: Construct probability distributions	★	★	★			★	★	★	★	★	★
PCS: Implement simple algorithms to generate synthetic data sets	★					★	★	★	★	★	★
TS: Understand and discuss core concepts in probability	★	★	★		★	★			★	★	★
MVD: Graph core concepts in probability	★		★			★	★	★	★	★	★
NQP: Illustrate the effect of parameter dependence on distributions					★		★		★		★

To keep your expectations and goals in the right place, these are the skills and concepts we hope you have gained in the course of reading these notes, completing all the worksheets, and completing the assignment. After this module, you should be able to:

1. Theory
   (a) Describe the concept of probability and random variables.
   (b) Interpret the notation $P(X = x)$.
   (c) Define and interpret the joint, conditional and marginal distributions of two random variables.
   (d) Discuss theoretical and empirical probability distributions and the differences between them.
   (e) Explain why $\int_{x \in X} P(X = x)\, dx = 1$
   (f) Determine when two random variables are independent.
   (g) Discuss the Central Limit Theorem and its assumptions. Describe the $\sqrt{N}$ rule.
   (h) Discuss Bayes' Theorem and identify the components of Equation 7
   (i) Discuss and interpret the Binomial Formula.
   (j) Discuss and interpret the definition of probability density functions and cumulative density functions.
2. Calculations
   (a) Given some data (or a theoretical distribution), calculate: $P(X = x)$, $P(X \leq x)$, $\sum_{x \in X} P(X = x)$.
   (b) Given data or a distribution, calculate $E[X]$, $Var[X]$, and other moments.
   (c) Given data or a distribution, calculate the median, IQR, or other percentiles.
   (d) Given data or a distribution, calculate the mode.
3. Visualize
   (a) Plot empirical or theoretical PDFs and CDFs.
   (b) Use vertical or horizontal lines to show means, medians, and percentiles on PDFs and CDFs.
   (c) Show how different quantities change as a function of (simulated) parameters.
4. Simulations and Coding

(a) Simulate coin tosses with arbitrary N and p.

(b) Implement Bayes' Theorem to calculate a posterior distribution.

(c) Extract time intervals from a time series (Assignment 1).

(d) Generate PDFs and CDFs from data.

In the worksheets and assignments you will implement and grapple with all of these ideas. Enjoy!

6 Other Details

6.1 More Probability Rules

Consider two random variables A and B. Let a and b be specific outcomes of A and B, respectively. The following are some mathematical rules that you can use to manipulate probabilities. These are absolutely not necessary to have memorized or accessible to be successful in this course, but thinking about them might help you gain some intuition about probabilities.

6.1.1 The Addition Rule

If we want to know the likelihood of either of two outcomes occurring, we can use the **addition rule**.

$$P(A{=}a \text{ OR } B{=}b) = P(A{=}a) + P(B{=}b) - P(A{=}a \text{ AND } B{=}b) \tag{9}$$

6.1.2 The Multiplication Rule

We have already seen this in the notes, but this rule is sometimes called the **multiplication rule**:

$$\begin{aligned} P(A{=}a \text{ AND } B{=}b) &= P(A{=}a) \times P(B{=}b \,|\, A{=}a) \\ &= P(B{=}b) \times P(A{=}a \,|\, B{=}b). \end{aligned} \tag{10}$$

Notice the symmetry in the way this relation can be written.

6.1.3 The Complement Rule

This should be an obvious consequence of the fact that $\sum_{x \in X} P(X{=}x) = 1$ (the likelihood of *something happening* is 100%), but this is formally called the **complement rule**.

$$P(A{\neq}a) = 1 - P(A{=}a) \tag{11}$$

6.1.4 Law of Total Probability

We can always break probabilities of one random variable into conditional statements that depend on other random variables. This is explicitly done with the **Law of Total Probability**.

$$P(A{=}a) = P(A{=}a \,|\, B{=}b) \cdot P(B{=}b) + P(A{=}a \,|\, B{\neq}b) \cdot P(B{\neq}b) \tag{12}$$

That is, the probability that $A = a$ can be found conditionally by considering how likely it is when $B = b$ and adding the likelihood that arises when $B \neq b$.

6.1.5 Marginal Distributions

This previous rule suggests that we can extract probabilities of one random variable from a joint distribution. Indeed, if we want to know the probability distribution of A without consideration of B, we can add up the way that A is distributed over all the different outcomes of B. This is called **marginalization** and will come up later in the course. More concretely, if we have a joint distribution $P(A, B)$, then we say that $P(A)$ and $P(B)$ are **marginal distributions** of A and B. We can find these quantities from the joint using a summation (or integral):

$$P(A) = \sum_{b \in B} P(A, B) \tag{13}$$

$$P(B) = \sum_{a \in A} P(A, B) \tag{14}$$

6.2 More Theoretical Distributions

For the sake of clarity and brevity we ignored introducing the specific functional forms of some relevant and interesting distributions. We list them here with some relevant notes. Again, it is not necessary to memorize any of these formulas, and we will remind you of any relevant details as we go forward.

As a note about notation, probability distributions generally have **parameters** that determine their shape and moments. When we have a distribution of some quantity, x, and it has parameters $\theta_1, \theta_2, \ldots$, we will generally notate this distribution as

$$P(x\,;\,\theta_1, \theta_2, \ldots) \qquad \text{OR} \qquad P(x\,|\,\theta_1, \theta_2, \ldots). \tag{15}$$

It may seem odd that we're re-using the notation for conditional probabilities, but you can convince yourself that the notation we're trying to express actually *is* a conditional probability – we're conditioning on the different parameters.

6.2.1 The Binomial Distribution

We have already discussed this extensively in the text, but the **binomial distribution** describes the distribution of the number of **successes**, k, that occur in N trials, where the likelihood of a single success is p. If a "success" is a coin coming up heads, then this distribution describes the number of heads in N flips of a coin.

The formula for the PDF is given by Equation 3, which we repeat here:

$$P_{Binom}(k\,;\,N, p) = \binom{N}{k} p^k (1-p)^{N-k}. \tag{16}$$

In the `scipy.stats` package, you can access the binomial distribution with the `scipy.stats.binom` object.

6.2.2 The Gaussian Distribution

Perhaps the most important distribution ever, the **Gaussian** or **Normal** distribution describes a wide variety of phenomena. We'll discuss it increasingly as the course progresses, but for now we'll just give some basic facts.

Normal distributions are parameterized by their mean, μ, and variance, σ^2, which should be interesting on its own. The PDF is given by

$$P_{Gauss}(x\,;\,\mu, \sigma^2) = \mathcal{N}\left(x\,;\,\mu, \sigma^2\right) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}. \tag{17}$$

When $\mu = 0$ and $\sigma^2 = 1$, the distribution is referred to as the **standard normal distribution**. You can convert any normal distribution to a standard normal by making the substitution $y = (x - \mu)/\sigma$ (subtracting the mean and dividing by the standard deviation).

This is known as **standardization**.

In the `scipy.stats` package, you can access the binomial distribution with the `scipy.stats.norm` object.

6.2.3 The Poisson Distribution

The Poisson distribution is another useful distribution to know about as it arises frequently in natural processes. It describes the distribution of events that occur in a given time interval, if the average number of events in an interval is λ (also known as the arrival **rate**), and each event occurs independently of the last. The Poisson distribution can be considered to be a limiting case of the binomial distribution when the probability of success is very small.

The PDF of the Poisson distribution is

$$P_{Poisson}(k\,;\,\lambda) = \frac{\lambda^k e^{-\lambda}}{k!} \tag{18}$$

In the `scipy.stats` package, you can access the binomial distribution with the `scipy.stats.poisson` object.

6.2.4 The Exponential Distribution

Finally, the **exponential distribution** describes random variables whose PDFs are exponential functions. You will explore the exponential distribution extensively in the assignments, so for now we will only say that the exponential distribution can be used to describe the time interval between Poisson-distributed events.

The PDF of the exponential distribution is

$$P_{Expon}(x\,;\,\lambda) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{19}$$

In the `scipy.stats` package, you can access the binomial distribution with the `scipy.stats.expon` object.

Chapter Three: Parameter Estimation and Model Fitting

Eric Johnson and Madhav Mani

May 11, 2020

In the previous chapter we introduced the basics of probability and hopefully convinced you that a probabilistic view of the world might be a useful framework in which to perform experiments and analyses. However, the critical reader may have noted that we didn't really gain much in the way of tools that let us *learn new things*. We covered probability distributions and the Central Limit Theorem (CLT), but you would be justified in feeling that you don't know how to use this new information practically. In this chapter, we will introduce the first two tools that a quantitative researcher should employ when attacking a data set:

- Estimating model parameters from data and creating bounds on where we expect the parameter to exist.
- Least-squares regression for fitting (linear) models.

Students who have seen the words "confidence intervals" and "linear regression" may be surprised that we're spending time on these topics, but what we hope to show you is that these two tools are much more profound and useful than one might expect. In particular, we will see how these methods will help us start to answer the question **"What do your data say?"**

It's also worth noting that these concepts form the basis of almost all advanced methods in statistics and machine learning, including dimensionality reduction, manifold learning, and clustering. The effort you put in to understand the basics of parameter estimation will help you use these tools more confidently and be more critical about when and how they are applied.

1 The Atom of Parameter Estimation

In this first section we're going to discuss what it even means to "estimate a parameter". In the previous chapter, we learned how to calculate moments and percentiles, but we didn't discuss how to answer the question "what number should I tell my neighbor that I measured?" That is, if I have to report one value for some quantity, what value should I give? This is called making a **point estimate** of this quantity.

Parameter Estimation Questions

Once we've gotten our point estimate, some natural questions are:

- How likely am I to observe this estimate?
- How confident are you in this estimate?
- How much deviation from this estimate should I expect if I re-did your experiment?

These questions are innately tied to a probabilistic and distributional way of thinking and can be answered by looking at the *shapes* of specific distributions. Specifically, we'll show how to calculate classical **confidence intervals**, Bayesian **credible intervals**, and how to use **bootstrapping** to reconstruct how our quantity of interest is distributed. In particular, by the end of this module, you should see that bootstrapping is one of the most useful data analysis tools we have for attacking data of any kind!

1.1 Parameter Estimation

Before we can get to the exciting parts about intervals and curve-fitting, it is worth spending a moment being precise about what we are even talking about when we say "estimate this quantity." To explore this, consider a set of N independent and identically distributed (i.i.d.) random variables x_i. Let's say that we're interested in the mean of these random variables, which we'll call μ. (For example, maybe we're counting the number of heads in some coin flips, or the number of photons incident on a detector, or we want the mean time that a bacteria spins counter-clockwise.)

Let's say that you've collected your data, the x_i, and then your collaborator comes by and asks for your estimate of the mean, let's denote it $\hat{\mu}$; what do you say? You may be inclined to say that your estimate is simply the average ($\bar{x} = \frac{1}{N}\sum_{i=1} x_i$), so you pull out a calculator and tell your collaborator that $\hat{\mu} = \bar{x} = 10$. This works great until a week

later you realize that you forgot to enter some of the x_i into your calculator and actually $\bar{x} = 9$. You run over to your collaborator who kindly re-runs their analysis with $\bar{x} = 9$, but you are left wondering what the *true* mean, μ, of your random variables is. That is, if you added a few more observations, how much would the mean change? What if you repeated the experiment, what mean would you calculate then? If you had infinite data, what would you calculate?

Concerned by this, you take some time and repeat your experiment to get a new set of N observations, x_j. You calculate $\bar{x}$ and get 12! You've now seen $\bar{x} = 9, 10$, and 12, so which should I use for $\hat{\mu}$?

To answer this question, let's recall that we can consider not only our data x_i to be random variables, but also consider our estimate for the mean, $\hat{\mu}$, to be a random variable. That is, we can try and describe our quantities, x_i and $\hat{\mu}$ via *probability distributions*, $P_x(X = x_i)$ and $P_{\hat{\mu}}(M = \hat{\mu})$. However, the problem arises in that while we might have many (N) data points, x_i, so that we can at least empirically describe P_x, we only have one value for $\hat{\mu} = \bar{x}$, so we can't make a good description of $P_{\hat{\mu}}$ empirically. Of course, if we *could* pin down $P_{\hat{\mu}}$, theoretically or empirically, then all of the work in the previous section would help us immensely: we could calculate the expected value of $\hat{\mu}$ and we could start to talk about the spread in $\hat{\mu}$ that we might see.

The Big Problem

The problem of describing $P_{\hat{\mu}}$ is what is called the problem of **parameter estimation**.

Going forward, we are going to continue trying to estimate the mean, μ, of a set of random variables, x_i, but it's worth noting that this problem generalizes to *any parameter*, θ, that might depend on our data. That is, while we're going to elaborate on some *specific* calculations for $\hat{\mu}$, we're also going to outline the *general* techniques for any parameter estimator $\hat{\theta}$.

1.2 Maximum Likelihood and Maximum a Posteriori Estimates

Before we get too much further, it should be said that there is no "correct" answer to the problem of "how should I calculate $\hat{\theta}$, an estimate of θ?" The Wikipedia page on point estimation lists *nine* different methods for answering this question, each of which uses its own assumptions and tools. What we want to highlight and emphasize are two of the most widely-used and intuitive methods, **Maximum Likelihood Estimation** (ML Estimation)

and **Maximum a Posteriori Estimation** (MAP Estimation).

1.2.1 Maximum Likelihood Estimation

The premise of ML estimation is simple: write down the **likelihood** function, $f(x_i|\theta)$, and say that $\hat{\theta}$ is the value of the parameter θ that *maximizes the likelihood*. $\hat{\theta}$ is known as the **maximum likelihood estimator** (MLE) for the parameter θ. In the case of a mean of i.i.d. random variables, thanks to the Central Limit Theorem (CLT) we know that μ is distributed as a Gaussian distribution with mean μ and standard deviation $\sigma = \sigma_x/\sqrt{N}$, where σ_x is the standard deviation of x_i. If this isn't clear, make sure to return to the previous set of notes for more details. The important point here is that the CLT tells us that *sums* are distributed normally.

Then ML estimation says that what we want to do is consider the likelihood of observing our data $\bar{x}$ given different values for μ and maximize it. The formula for a Gaussian distribution is

$$P\left(z \mid \mu, \sigma^2\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(z-\mu)^2}{2\sigma^2}\right), \tag{1}$$

where z is the quantity that is distributed, μ is the mean of z, and σ^2 is the variance of z.

Try It Yourself

In Python, create a variable for the mean and variance, `mu` = 2 and `sigmaSq` = 1. Create a grid of z-values (maybe using `np.linspace`) and implement this formula to calculate the probability as a function of z. Use `plt.scatter` to show the calculated probabilities versus the z-values and confirm that you see the expected "bell" curve!

In the present case, we are considering the distribution of the mean, $\bar{x}$, so we will substitute $\bar{x}$ for z in this equation. We then want to find the value of μ that maximizes $P(\bar{x} \mid \mu, \sigma^2)$, which we write in symbols like this:

$$\hat{\mu} = \max_{\mu} \left[\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(\bar{x}-\mu)^2}{2\sigma^2}\right)\right] \tag{2}$$

This notation says that we should set $\hat{\mu}$ to be the value of μ (as indicated by the μ under the "max") that maximizes the quantity in the square brackets. We can then take the logarithm of the stuff inside the brackets and drop the $1/\sqrt{2\pi\sigma^2}$, as this won't change where

the function is maximized.

$$\hat{\mu} = \max_{\mu} \left[\log \left[\exp \left(-\frac{(\bar{x} - \mu)^2}{2\sigma^2} \right) \right] \right] = \max_{\mu} \left[-\frac{(\bar{x} - \mu)^2}{2\sigma^2} \right]. \tag{3}$$

We can then quickly see that $(\bar{x} - \mu)^2$ is always positive, so this function is maximized when $\bar{x} = \mu$ and we can write that the MLE is

$$\hat{\mu} = \bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i \tag{4}$$

because setting $\hat{\mu}$ to this value *maximizes the likelihood* of observing the data x_i.

If the math here is giving you difficulties, don't worry; it's here for the sake of being thorough, not because it's essential to doing parameter estimation. The thing you should take away is that *one way* of making estimates is to maximize a likelihood function. Also, as the "Try It Yourself" boxes will try to point out, if you don't understand a formula or equation on paper, you can always try and *see* it in your computer with Python.

Try It Yourself

Again using `mu` $= 2$ and `sigmaSq` $= 1$, plot the quantity from Equation 3

$$-\frac{(z - \mu)^2}{2\sigma^2}$$

versus z alongside your bell curve from above. Where is this quantity maximized? Is it at the same place as the Gaussian distribution? Does changing σ^2 affect anything?

In general, given some **model** connecting x_i and your parameter θ, which we'll call $f(x_i|\theta)$, then we can make an MLE, $\hat{\theta}$, by maximizing this function over θ. In this case, it turns out that our gut call of using $\hat{\mu} = \bar{x}$ was equivalent to using the maximum-likelihood estimator; this will not always be the case!

We'll continue to make MLEs later, but for now we'll summarize the idea as so:

Maximum Likelihood Estimation

Using a **likelihood** function $f(\vec{x}|\theta)$, the **Maximum Likelihood Estimator** $\hat{\theta}$ is defined as $\hat{\theta}$:

$$\hat{\theta} = \max_{\theta} \left[f\left(\vec{x} \,|\, \theta\right) \right]$$

We'll show in a moment that we can also make an MLE for the **variance**, σ^2, of the distribution of the mean, $\bar{x}$. If you're uncomfortable with calculus, please feel free to skip over the following example to the formula at the end. What you should take away from this example and the one above is that ML estimation is often an exercise in *algebraic manipulation*: you take your likelihood and find where it is maximized. Once you have the formula, you're good to go. However, we caution that if you *can't* follow or assess the derivation, then you should be skeptical that you understand the formula well enough to use it on its own, and you should bolster your conclusions by making the estimate in a different way as well.

Example: Estimating the Standard Deviation

First, for this estimate, we're going to start by *assuming* that our data, x_i, are distributed normally, with mean, μ, and standard deviation σ_x. Then the likelihood of observing a specific point, x_i, is given by subsituting $x_i = z$ into Equation 1. We furthermore assume all of our observations are independent so that the likelihood of all the points is the *product* of their individual likelihoods. That is:

$$P\left(x_1, x_2, x_3, \dots \,\middle|\, \mu, \sigma_x^2\right) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma_x^2}} \exp\left(-\frac{(x_i-\mu)^2}{2\sigma_x^2}\right) \tag{5}$$

$$= \left(\frac{1}{2\pi\sigma_x^2}\right)^{N/2} \exp\left(-\frac{1}{2\sigma_x^2}\sum_{i=1}^{N}(x_i-\mu)^2\right), \tag{6}$$

where $\prod$ is the notation for a **product** of terms that increment i from 1 to N and the product of exponentials sum in the exponent.

We want to find the value of σ_x that maximizes this function, so we're going to use a bit of calculus to do so (this problem isn't as obvious as finding the estimate for μ). First, however, we'll again use the fact that maximizing the *logarithm* of the likelihood (often called the **log-likelihood**) is the same as maximizing the likelihood itself. This lets us write

$$\log\left[P\left(x_1, x_2, x_3, \dots \,\middle|\, \mu, \sigma_x^2\right)\right] = -\frac{N}{2}\log\left[2\pi\sigma_x^2\right] - \frac{1}{2\sigma_x^2}\sum_{i=1}^{N}(x_i-\mu)^2 \tag{7}$$

Taking the derivative with respect to σ and setting it equal to zero lets us find the

maxima of this quantity so that we can write

$$0 = -\frac{N}{\sigma} + \frac{1}{\sigma^3}\sum_{i=1}^{N}(x_i - \mu)^2 \qquad \Rightarrow \qquad \sigma_x^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2. \tag{8}$$

Now, we're not quite done yet! This is an estimate for the standard deviation of the *data*. We said that we wanted an estimate for the standard deviation of the *mean of the data* (since it's a random variable). However, if you try to do what we just did with the likelihood of the mean in Equation 2, you will encounter

$$\hat{\sigma}^2 = (\bar{x} - \mu)^2, \tag{9}$$

but if we don't know μ (which we often don't!), then using our best guess that $\hat{\mu} = \bar{x}$ tells us that $\hat{\sigma}^2 = 0$! This is obviously incorrect, so to get even somewhere close to an answer, we had to assume that the data were normally-distributed.

So now we've found that our best estimate for the standard deviation of normally-distributed random variables is given in Equation 8. Note this formula takes the form we'd expect: it's the definition of how we were taught to calculate the standard deviation! (Of course, this is a biased estimator of the standard deviation, but we'll leave that discussion to other resources.) To get an estimate for the standard deviation of the *mean*, we'll invoke the CLT, which tells us that the mean's variance is σ_x^2/N so that we can use

$$\hat{\sigma} = \frac{\hat{\sigma}_x}{\sqrt{N}} = \frac{1}{N}\sqrt{\sum_{i=1}^{N}(x_i - \hat{x})^2} \tag{10}$$

as our estimate.

Again, the point of this example was not to prove that we can do math but to demonstrate what ML Estimation looks like in practice. There's nothing particularly profound about it, but the formulas you learn are sometimes more constrained by algebra than whether they are really what you want to estimate. That is:

You can sometimes maximize the likelihood function to estimate parameters.

Of course, if we're not concerned with doing any algebraic analysis of the MLE, then we can always use computational techniques to find the maximum of our likelihood function. In the examples shown in these notes, where we only have a few parameters, this is probably doable, but in more complicated scenarios can be quite difficult. In this way, while ML estimation is a hugely successful methodology for estimating parameters, as you saw above and will see in the assignments, actually determining closed form answers for an estimator sometimes be tricky. In particular, for those not fluent in probability notation and calculus, bringing the simple premise of "maximize this function" to life can be quite difficult! As a result, statistics textbooks become complex lists of recipes, because it takes so much work to derive the results that it can't be asked of the average user.

Thankfully, as we noted, there are alternative approaches that are sometimes more effective or practical. In the next section we'll introduce **Maximum *a Posteriori* Estimation** (MAP Estimation), which still relies on mathematical theory to work, but in a more flexible manner, and later we'll introduce the computational technique of **bootstrapping**.

1.2.2 Maximum *a Posteriori* Estimation

MAP estimation is a method for generating point estimates in a manner similar to MLE except that instead of maximizing the likelihood function, we find the maximum of the *posterior* distribution for our parameter (hence the name maximum *a posteriori*). Consider Figure 1, where the Bayesian posterior generation process for the mean of some data has been illustrated.

In this Figure, we've posed two priors: a uniform prior for the mean, μ, from -1 to 1 and a Gaussian prior with a shifted mean and widened variance.

$$P(\mu) = \mathcal{U}(-1, 1) \qquad \text{and} \qquad P(\mu) = \mathcal{N}\left(\bar{x} - 0.4, 1.5 * \sqrt{\frac{Var[x]}{N}}\right) \tag{11}$$

If we use the MLE for the standard deviation, the likelihood function for the mean is the same as in Equation 2:

$$P\left(\bar{x} \,\middle|\, \mu, \sigma = \sqrt{\frac{Var[x]}{N}}\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(\bar{x} - \mu)^2}{2\sigma^2}\right] \tag{12}$$

We can then multiply these distributions to get the posterior computationally.

Thanks to the modern computer, making both ML and MAP estimates can be as simple as finding the maximum value of a curve. What is nice then about using a Bayesian

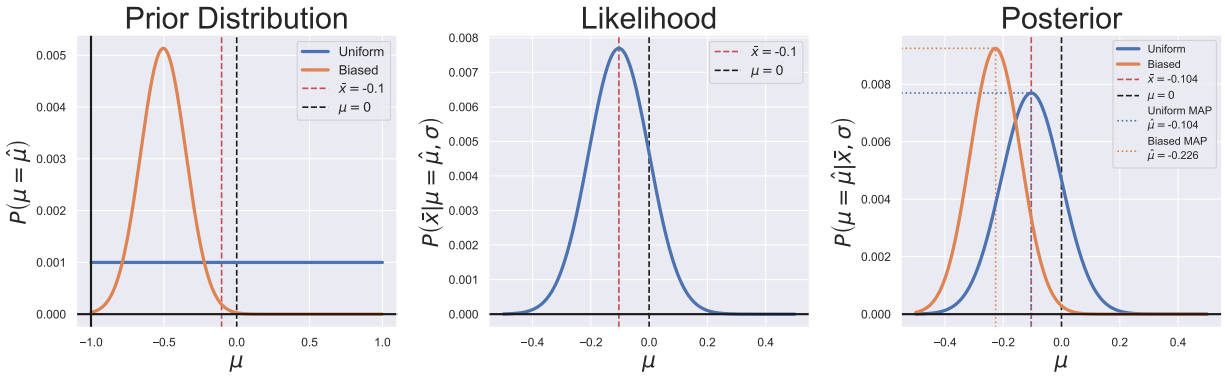


Figure 1: Illustration of the MAP estimation of the mean of $N = 100$ normally distributed random variables under two different priors. All three panels show the location of the MLE ($\hat{\mu} = \bar{x} = -0.104$) and the true population mean, $\mu_{TRUE} = 0$. The left panel shows two different prior distributions for the sample mean, μ. The center panel shows the likelihood function for the data (σ was set to be $\sqrt{\text{Var}[x]/N}$). The right panel shows the two posteriors and their MAP estimates.

framework here is that we are explicitly incorporating any priors that we want and we don't need to do too much theoretical work. Moreover, we bring up the Bayesian method because it more naturally facilitates talking about the whole distribution instead of a single point estimate. In fact, because the point of a Bayesian analysis is to end up with a distributional description, researchers often use the posterior expected value or **posterior median** as their point estimate. The posterior median is often preferred since it is less sensitive to outliers or bad priors, as can be seen in Figure 2, where a bad prior caused the posterior to be multi-modal (have multiple peaks).

We also bring up the Bayesian approach because it makes clear some of the implicit assumptions that often accompany an MLE. For example, the likelihood function and a uniform prior gives a posterior that is identical to the likelihood, so that the MLE and MAP estimate will be the same. This can clearly be seen in the blue lines in Figure 1, where the likelihood and posterior functions are both normal distributions that are centered on our MLE $\hat{\mu} = \bar{x}$. In this way, the Bayesian approach is somewhat more flexible than the MLE approach in that we can loosen some assumptions on the parameters.

In any case, hopefully you now have some idea of the choices we can make when giving our collaborator just one number to describe parameters. When we have some theory about how a parameter is distributed, we can try to make an MLE. If we have a prior model for the parameter and a likelihood model for the data, then we can make posterior-based estimates.

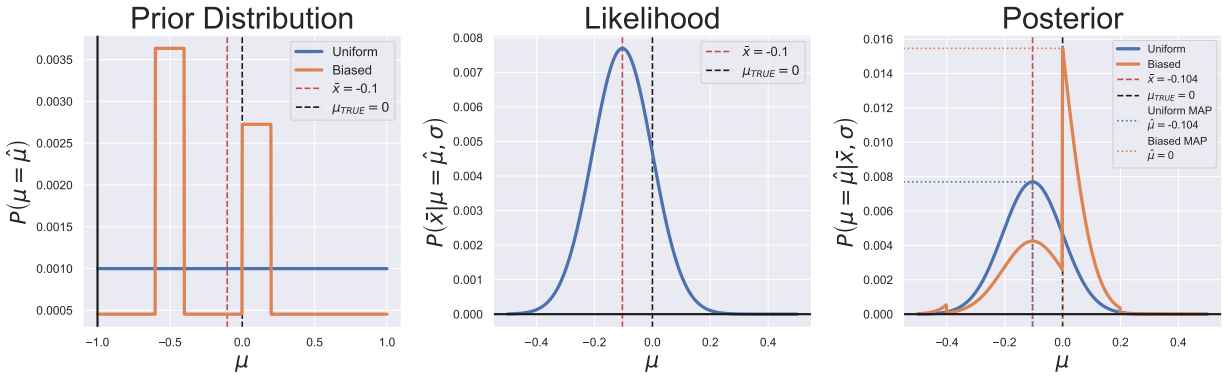


Figure 2: Illustration of the MAP estimation of the mean of $N = 100$ normally distributed random variables under two different priors as in Figure 1.

Takeaway: MAP Estimates

MAP estimation is similar to ML estimation in that it is dependent on some theory in order to work. However, incorporating priors allows for greater flexibility in assumptions.

For a more detailed example of how to make a MAP estimate of quantities derived from exponentially distributed data (for example, from the time intervals from the bacterial chemotaxis data), look to the end of the notes. You will use the code and results of that example in Assignment 2!

Try It Yourself

Use your Assignment 1 solutions or those provided to generate a MAP estimate of the switching rate λ^+ or λ^- by maximizing the posterior distributions *computationally*. That is, implement a solution and find the value of λ at which the posterior is maximized. Compare this value to the posterior mean, median, and the MLE.

1.3 Intervals of Confidence

If you're following closely, you may have noted that the previous section, while elaborating on topics from the first chapter, still didn't answer all of our questions:

- How likely am I to observe this estimate?
- How confident are you in this estimate?
- How much deviation from this estimate should I expect?

This is intentional; it helps to ground yourself more specifically in the problem at hand before we introduce the nebulous quantity of **confidence**. Recall your earlier experiments with x_i and x_j and how you measured $\bar{x}$ to be 9, 10, and 12, depending on which data was included in the calculation. We also discussed how depending on what type of estimator you decide to calculate, you might report different numbers. So the path forward might still seem a bit murky.

One potential solution is to model the sample distribution for the parameter (say, $\hat{\mu}$) and report that the MLE $= \bar{x} = 10$ has a likelihood of 0.12, so that we can answer our first question. But turning to the second, that answer might be concerning: 88% of the time it is not that number! How can that be our report? How can we have confidence in something that only happens 12% of the time?

The answer, as you have probably started to pick up on, is to use the probabilistic and distributional nature of these quantities to look not just at single points, but also to look at probabilities of whole *regions* of parameter space. That is, rather than saying "$\hat{\mu} = 10$", we'd like to say things like, "I expect $\hat{\theta}$ to be in this interval 82% of the time" or "60% of new experiments will deviate from $\hat{\theta}$ by 2 or less." In particular, this section will work on the development of specific types of intervals: **confidence intervals**, **credible intervals**, and finally, the use of **bootstrapping** to generate confidence intervals.

Before we get too much further, it's also worth commenting on what we mean by *confidence*. Intuitively, it is useful to think about the phrase "I have ____% confidence in this parameter having these values," so that "confidence" becomes synonymous with "probability of observation." This is a useful intuition as it lets us quantify a desirable outcome of our analysis. However, as we'll explain, the exact way that statement this will manifest itself will depend on how we decide to construct our intervals.

1.3.1 Confidence Intervals

First, and most briefly, we need to discuss confidence intervals. Most directly, confidence intervals describe a region into which the true parameter value falls into $1-\alpha\%$ of the time. (α is known as the **confidence level**) This is a precise and somewhat confusing statement, so it bears rephrasing: if you were to perform an experiment 100 times and measure the parameter estimate, $\hat{\theta}$, and the 80% confidence interval of each of those experiments, then we would expect 80 of those confidence intervals to contain the true population parameter, θ. This website provides an excellent visualization of how this works.

These comments are necessary before getting into the *how we calculate* confidence intervals because this is a non-obvious part of these tools. Confidence intervals represent the range I would need to report to cover myself from being disagreed with in $1-\alpha\%$ of future experiments. This is a very different quantity than the confidence we'll be assessing with credible intervals.

OK, now for the *how* of confidence intervals. Typically, as with ML estimation, one has to know something about how the parameter of interest is distributed, then intervals can be worked out based on that parameterization. For example, the sample mean, $\bar{x}$ is distributed normally, so we can say that it has a confidence interval given by

$$\left[\bar{x} \pm z_{\alpha/2}\frac{s}{\sqrt{N}}\right], \tag{13}$$

where $z_{\alpha/2}$ is the $\alpha/2^{\text{th}}$ percentile of the standard normal (z) distribution, and s is the sample standard deviation $\frac{1}{N-1}\sum_{i=1}^{N}(x_i-\bar{x})^2$. This is because this is the region in which $\alpha\%$ of the probability of a normal distribution lies.

That is, in general, we calculate confidence intervals by determining upper and lower bounds for a region such that some fraction of a distribution's probability falls in that region. The amount of probability that we want in the region is called the *confidence level*, and is typically indicated with $1-\alpha$. Common values for α are 0.1, 0.05, or 0.01 (90%, 95%, or 99%), depending on the field of research and type of data.

Try It Yourself

Use this formula to add on to the work you did in Worksheet 1.2. Specifically, how does the confidence interval change as you change N, M, and p?

More formally, if we have a parameter, θ, then we want to solve

$$1 - \alpha = P\left(L \leq \theta \leq U\right) = \int_{L}^{U} P\left(\theta = t\right) dt \tag{14}$$

for L and U, our lower and upper bounds. Equation 13 is the result of solving this for the case when $\theta = \mu$, the mean of a set of random variables, so that $P(\theta)$ is a normal distribution. But for other parameters that are distributed differently, like a Poisson rate parameter or a regression coefficient, we'll need to solve this anew. It often works to approximate the confidence interval with Equation 13, which is likely why it seems intuitive to use `mean`$\pm$`standard deviation` as a confidence interval of sorts, but it doesn't work frequently or generally enough to be our only tool. At the end of these notes we'll delve into the derivation of the confidence interval for an exponentially rate parameter that you will make use of in the assignment.

Indeed, it is precisely because of this specificity that we move on to other methods. It is not to say that knowing about confidence intervals is not useful, just that keeping a list of theoretical derivations is not generically useful enough to be the only weapon in your arsenal.

1.3.2 Credible Intervals

An alternative approach is the Bayesian **credible interval**, which is an interval of "confidence" that we can generate from a posterior distribution. Specifically, if we have a posterior distribution for how our parameter of interest, θ, is distributed, then we can use the usual interpretation of a probability distribution to find bounds around $\hat{\theta}$, our estimate, the enclose $1 - \alpha\%$ of the probability (mass) of θ occurring. That is, rather than worrying about exactly how θ is distributed theoretically, we can consider our posterior and find a region that contains 95% of the probability (if $\alpha = 0.05$).

To do this, we have a few options, but we'll only discuss the easiest: calculate the CDF and find the $\alpha/2^{\text{th}}$ and $(1 - \alpha/2)^{\text{th}}$ percentiles. The interval between these two points is a region in which θ has a likelihood of $(1 - \alpha)\%$ of existing.

Note that the use of this interval is slightly different than that of a confidence interval in that we are directly considering θ to be a random variable and we are discussing the probability of θ existing in a certain region. Confidence intervals estimated our ability to know the true value of θ, which is not a random variable, and therefore are useful in the context of setting expectations for further experimentation. The credible interval describes

our confidence *right now* about where *theta* lies.

Again, when we use a uniform prior on our parameter θ, then the credible interval should coincide with the confidence interval. The main benefit then of using a Bayesian approach is basically that if we're going to be computationally generating the confidence interval anyways, then we might as well explicitly indicate what prior beliefs we have. Always assuming that the parameter is uniform is clearly not a robust or safe assessment!

Finally, we note that while taking the interval given by the $\alpha/2^{\text{th}}$ and $(1-\alpha/2)^{\text{th}}$ percentiles gives an interval with the correct amount of probability, it is not the *only* interval that gives this amount of probability. For example, at a confidence level of 90%, we could have chosen the 1^{st} and 91^{st} percentiles, or the 7^{th} and 97^{th}. Given this, it is sometimes recommended to find the interval containing $1-\alpha\%$ probability that is the *smallest*. This is called a **highest density region** or HDR. For most unimodal (only one peak) posteriors the HDR is very similar to the difference between the $\alpha/2^{\text{th}}$ and $(1-\alpha/2)^{\text{th}}$ percentiles, but for multimodal distributions, the HDR might be more intuitive, as explained in this post.

In any case, the technical differences between using an HDR or percentiles are secondary to the probabilistic interpretation of the credible interval as a region where I think θ exists now. Confidence intervals, which describe how well I think I can measure θ in any given experiment, are a different beast.

1.3.3 Bootstrapping

Given the difficulties with both of the previous methods (confidence intervals are theoretically difficult and credible intervals can be computationally difficult), you may be feeling a bit worried about how hard it is to answer "where is θ!?" Thankfully, the modern computer has given us the ability to use a computational technique known as **bootstrapping**.

As described somewhat comprehensively in 1986(!!!), Tibshirani and Efron show that if we have some data, x_i and we want to estimate some parameter θ. We can *approximate* the true distribution of θ by **resampling** our data **with replacement** and recalculating θ on our resampled datasets.

That is, I take my data $x_1, x_2, \ldots, x_N$ and I make a new data set of N points by randomly drawing data points from my collection x_i, where each random draw comes from the full set of x_i, so that my new data set may have multiple x_2 and x_{17} (for example). This is what is meant by *with replacement*; if we were drawing our cards from a deck, I would be *replacing* the cards I had drawn so the deck is always full.

Using this new data set, I calculate $\hat{\theta}$. Then I generate a new data set and do it again.

Then I do the process again many times. Eventually I will have a *distribution* for $\hat{\theta}$ and it can be shown that this distribution will eventually (with resampling) become equivalent to the sampling distribution that I was trying to describe with a confidence interval. So instead of performing a bunch of theoretical calculations, I can instead look at the percentiles of the bootstrapped distribution for $\hat{\theta}$ to give my confidence interval, regardless of whether I know the exact way of writing down the distribution theoretically.

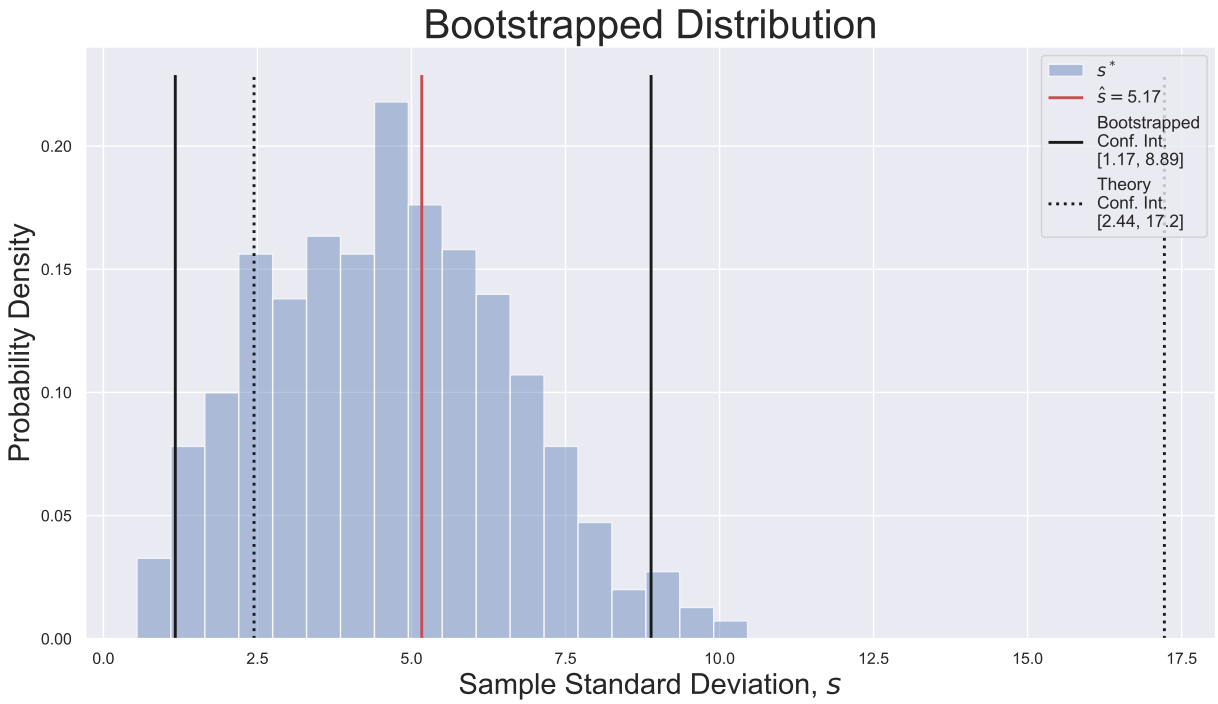


Figure 3: Bootstrapped distribution for the sample standard deviation (Equation 15). The original sample's estimate $\hat{s}$ is shown, as are the theoretical and bootstrapped confidence intervals. Note that the theory here does not match well with the bootstrapping because the random variables x_i were actually uniformly distributed, not normally distributed, which may have been hard to assess with only $N = 10$ samples!

To be extremely explicit, consider the following example where I have $N = 10$ observations

$$\vec{x} = [x_1, x_2, \ldots, x_{10}],$$

and I calculate their standard deviation

$$s = \frac{1}{N-1}\sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2}. \tag{15}$$

(The value I've calculated above is actually known as the **unbiased** estimator for the stan-

dard deviation, but that is a detail.) I want to know the confidence interval for s, but I don't know how theoretically how it's distributed, so I'm going to use bootstrapping. (For normally distributed data, s is distributed according to the aptly named Standard Deviation Distribution.)

To do this, I generate a new list of observations by randomly picking from $\vec{x}$. For example,

$$\vec{x}_1^* = [x_6, x_3, x_9, x_4, x_4, x_{10}, x_9, x_5, x_9, x_5],$$

where I can get multiple of one of my initial values because each time I pick, I pick from the full list. (As a quick hack, I generated these numbers in Python using

```
1 newIndices = (np.random.rand(10)*10).astype(int) + 1
```

See if you can make sense of this for your own use! Or ask about it in class.)

Using $\vec{x}_1^*$, I then use Equation 15 to calculate a new standard deviation s_1^*. I then repeat this process N_{BOOT} times: I generate $\vec{x}_1^*, \vec{x}_2^*, \ldots, \vec{x}_{N_{BOOT}}^*$ (we can collect these into an $N_{BOOT} \times N$ array called X^*). And I calculate $s_1^*, s_2^*, \ldots, s_{N_{BOOT}}^*$ and I *plot their distribution*. If I'm interested in an $\alpha = 5\%$ confidence level, then I can find the empirical 2.5^{th} and 97.5^{th} percentiles and report that 95% of the time, $\hat{s}$ falls in that region.

Bootstrapping Confidence Intervals

Given N data points $\vec{x}$, and a parameter $\theta(\vec{x})$, we can generate confidence intervals at a confidence level α with the following process:

1. Generate N_{BOOT} resamplings of $\vec{x}$ with replacement: $\vec{x}_j^*$.
2. Calculate $\theta^* = \theta(\vec{x}_j^*)$ for $j = 1, \ldots, N_{BOOT}$.
3. The confidence interval is the $\alpha/2^{\text{th}}$ and $(1-\alpha/2)^{\text{th}}$ percentiles of the distribution of θ^*.

Hopefully you can see that this is a relatively easy technique to use, it can be straightforwardly applied to many data sets, and its results are very interpretable, making bootstrapping an immensely useful tool.

Try It Yourself

At this point you should have everything you need to attempt Worksheet 2.1.
Once you've completed that, you should be able to start on Assignment 2, Problem 1.

2 The Atom of Model Fitting: Least-Squares Regression

At this point, hopefully you can see that we have at least attempted to answer our questions:

- How likely am I to observe this estimate?
  Answer: I can read off the height of the likelihood, posterior, or empirical bootstrapped distribution at that value to get this answer.
- How confident are you in this estimate?
  Answer: Using a confidence or credible interval, I can describe regions of confidence.
- How much deviation from this estimate should I expect?
  Answer: Confidence intervals give me bounds that overlap the underlying truth $(1-\alpha)\%$ of the time. Credible intervals tell me where $(1-\alpha)\%$ of the probability of observing θ is, given my data.

Furthermore, we now have the tools to really get into something interesting: **model fitting**.

In all of science, we are generically trying to build *relationships* between different phenomena. When I increase the volume of a gas, what happens to its temperature? If a neuron fires in one location of your brain, do you subsequently kick your leg? If I place a chemical gradient over some bacteria, which direction do they move? To do this, we use models, which can be extremely descriptive or very generic.

Why do we use these models? This is a somewhat deep question that many people have different answers for, but our answer is practical. The point of having a model is twofold:

1. A good model allows you to *extrapolate* beyond the regimes spanned by your measurements and to make new *predictions*.
2. A robust model that performs well in a variety of situations might be hinting at an underlying *principle* that can bring predictive understanding to a broader range of phenomena.

Either of these reasons would be sufficient justification for using modeling as a part of the scientific method, but it is worth noting that both are powerful and useful statements.

These models usually have *parameters*, which are quantities that often have some physical significance, and these parameters themselves need to be measured or estimated

in order for the model to make sense. This is the problem of model fitting: I think that I have the correct *functional shape* for my relationship (my quantities are related linearly, or logarithmically, or some other way), and I want to use my data to estimate the parameters of the model. In this way, you can see why we first talked about parameter estimation; model fitting is a specific application of this concept!

2.1 Ordinary Least-Squares Regression

In the following section we will present the method of **ordinary least-squares** (OLS) linear regression. This may seem to some readers to be a trivially simple method, but we hope that our discussion serves as a useful guide to ideas that generalize to more advanced methods that you might encounter. In this way, we'll be using linear regression as an example on which we can explore the concepts of model-fitting capability, under- and over-fitting, and predictive ability.

There are many ways to go about model fitting, but one of the most robust, famous, useful, and simplest methods is the method of **least-squares regression**. In the simplest version of this method, we have N pairs of data points (x_i, y_i) and we want to build a relationship between them. For this example, call x the *independent* variable and y the *dependent* variable; that is, we're going to model y as depending on x. In the context of regression, we often call x the **covariates** and y the **response**. Let's then pose the model $f(\vec{x}, \Theta)$, where Θ are the parameters of the model. We don't expect our model to be immediately (or really, ever) perfect, so let's define the **residual** to be discrepancy between the model and the data:

$$r_i = y_i - f(\vec{x}_i, \Theta).$$

The method of least-squares then seeks to find the values of the parameters Θ that *minimize* the sum of the squared residuals:

$$SS = \sum_{i=1}^{N} (r_i)^2.$$

This may seem somewhat technical, but it can also be observed graphically. Consider Figure 4. In this figure, we have data points (x_i, y_i) and we want to fit a line through this data. The method of least-squares considers the vertical deviations of the line from each y_i (the residual) and tries to minimize them.

That is, the method of least-squares suggests that the "best" line through the data is

Figure 4: Least-squares fit of a linear model, shown in red. The data to which the model is fit are shown in blue.

the one that generates the *smallest average (squared) residual* for any given data point.

It is worth noting that this is not the only way that we could have defined "best." We could have used the absolute value of the residuals; the squaring penalizes larger residuals more than smaller ones. We could use a weighted sum of the squared residuals because we want the line to fit certain data points better than others. We could redefine the residual to be a different distance. There are endless choices, and indeed many of these other choices have been explored by statisticians and scientists. The thought here is not that you need to always consider the entire spectrum of choices (obviously not, since most people don't consider anything other than OLS), but just to point out that this is a *choice* that you as an analyzer can make. One of the goals of this course overall is that regardless of how you make this choice, you will be able to assess your confidence in the outcome, and based on that you can make different choices.

That being said, there are some good reasons to use this method. First, the democracy with which the data points are treated is the best way to examine new data set. That is, I don't know *a priori* which data points are more or less important to my model, so I should treat them all the same as in OLS.

There is also a practical reason for using least-squares. The mathematical and compu-

tational algorithm for calculating the best-fit parameters is incredibly straightforward and fast. Before computers this mattered a lot, and even now, it makes more expensive techniques like bootstrapping and Markov Chain simulation that much more tractable if they can use fast algorithms many times.

The final reason is historical. Most advanced techniques in model fitting are based on modifications to least-squares. In order to understand more complex methods, understanding the strengths and weaknesses of ordinary least-squares (OLS) is a prerequisite.

2.1.1 Maximum-Likelihood Formulation

So how then do we actually solve this problem? In this subsection we'll outline the math of the OLS solution for **linear regression**. Later we'll talk about non-linear models, but it is instructive to consider the linear case first.

As a reader, you should not worry about following all of the math perfectly, but should try and establish some intuition about the results.

Lets start with the simple scenario where we have data about a single response variable and a single covariate. The classical way is to postulate the following:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i, \qquad \text{where} \qquad \varepsilon_i \sim \mathcal{N}(0, \sigma^2). \tag{16}$$

That is, y depends on x *linearly* plus some noise, ε that is **Gaussian distributed** with mean $= 0$ and variance σ^2.

This lets us think about y_i as a random variable with a mean that depends on x_i linearly. Specifically:

$$y_i \sim \mathcal{N}\left(\mu = \beta_0 + \beta_1 x_i, \sigma^2\right), \tag{17}$$

so that we can write the likelihood of observing any given y_i

$$P\left(y_i \left| x_i, \beta_0, \beta_1, \sigma^2\right.\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{\left(y_i - (\beta_0 + \beta_1 x_i)\right)^2}{2\sigma^2}\right] \tag{18}$$

$$= \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{\left(y_i - \mu_i\right)^2}{2\sigma^2}\right] \tag{19}$$

$$= \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{\left(r_i\right)^2}{2\sigma^2}\right] \tag{20}$$

Assuming then that each (x_i, y_i) pair is independent, the likelihood of observing all of

$\vec{y} = [y_1, y_2, \ldots, y_N]$ can be written as the product of the individual likelihoods.

$$P\left(\vec{y}\,\middle|\vec{x}, \beta_0, \beta_1, \sigma^2\right) = \prod_{i=1}^{N} P\left(y_i\,\middle|x_i, \beta_0, \beta_1, \sigma^2\right) \tag{21}$$

$$= \left(2\pi\sigma^2\right)^{-N/2} \exp\left[-\frac{1}{2\sigma^2}\sum_{i=1}^{N}\left(y_i - (\beta_0 + \beta_1 x_i)^2\right)\right] \tag{22}$$

$$= \left(2\pi\sigma^2\right)^{-N/2} \exp\left[-\frac{1}{2\sigma^2}\sum_{i=1}^{N}\left(r_i\right)^2\right] \tag{23}$$

Since $\frac{\sum_{i=1}^{N} r_i^2}{2\sigma^2}$ is always positive, you can see then that maximizing the likelihood is *equivalent* to minimizing the sum of squared residuals! We wrote this in the context of a linear model, but it's worth noting that this is true for any model $y_i = f(x_i, \beta)$.

OLS is Maximizing Likelihood

The solution to an ordinary least-squares problem is equivalent to finding the maximum likelihood estimator. (Assuming that the noise is **Gaussian distributed**.)

We've emphasized this in the text above, but we want to point out that many classical results in probability, statistics, and machine learning are predicated on some of the variables being normally distributed. While this is not a terrible assumption in the grand scheme of things, it does present difficulties when you aren't sure that your problem fits these assumptions. The reason that we don't stop the text here and tell you to use boilerplate algorithms is because there are *other useful ways* to make probabilistic statements about that data without resting on these assumptions. In any case, for the purposes of concreteness, we're going to see where the Gaussian approach can take us for a while longer.

Continuing on, just as we did in the section on ML estimation, we can use calculus to find the maximum of Equation 22 with respect to β_0 and β_1, our regression coefficients. Again, don't worry if you can't parse the algebra and notation, the takeaway here is that we can turn the crank of ML estimation to get formulas for the β that maximize the likelihood (minimize the sum of squared residuals), given our assumptions about the noise being Gaussian.

In the case of our single covariate linear regression (Equation 16), we can derive the following:

$$\hat{\beta}_0 = \frac{(\sum y_i)(\sum x_i^2) - (\sum x_i)(\sum x_i y_i)}{N\sum x_i^2 - (\sum x_i)^2} \quad \text{and} \quad \hat{\beta}_1 = \frac{N\sum x_i y_i - (\sum x_i)(\sum y_i)}{N\sum x_i^2 - (\sum x_i)^2}, \tag{24}$$

where the $\sum$ is notation that indicates a sum over terms that iterate $i = 1, \ldots, N$. ($\sum x_i = x_1 + x_2 + \cdots + x_N$).

This may seem complicated, but we can streamline this using linear algebra and matrix notation to write

$$\hat{\vec{\beta}} = \left(X^T X\right)^{-1} X^T \vec{y}, \tag{25}$$

where

$$\vec{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}, \qquad X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_N \end{bmatrix}, \qquad \text{and} \qquad \vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

Computers are *very good* at matrix operations, so this computation is very fast. Notice also that this notation allows us to generalize to as many covariates as we'd like by adding more columns to X and more rows to $\vec{\beta}$.

OLS Linear Regression

The solution to the least-squares problem for a linear model can be found with the formula:

$$\hat{\vec{\beta}} = \left(X^T X\right)^{-1} X^T \vec{y} \tag{26}$$

To understand this formula more concretely, consider the example at the end of these notes.

Try It Yourself

At this point, you should be able to attempt Worksheet 2.2.
Once you've finished that, you should be able to start the rest of Assignment 2.

2.1.2 Residual Distributions

You may have noticed a potentially large assumption in the previous subsection: the likelihood function depended on the fact that the noise, ε_i, was Gaussian-distributed with a *constant* variance σ^2. It turns out the that the OLS solution in Equation 26 doesn't depend on this assumption of Gaussianity, but the interpretation of the estimate as a maximum likelihood estimator, does. Furthermore, and more importantly, the theoretical derivation of the confidence intervals for our estimates $\hat{\beta}$ will depend on this assumption. As you'll see, this makes a tool such as bootstrapping even more useful, as it will generate confidence

intervals regardless of the underlying noise distribution.

Besides assuming that the noise is normally-distributed, the OLS solution also depends on the assumption that the variance in the residuals is *constant*, which is known as **homoscedasticity**. Mathematically, this appears in the fact that the size of the noise, σ^2, in our posing of the problem (Equation 16) is constant and doesn't depend on x_i (otherwise we might have said $\epsilon_i = \mathcal{N}(0, \sigma_i^2)$). But what this means in real terms is that we're assuming the noise is the same over the whole range of our measurements.

For example, imagine that you are measuring the relationship between temperature and humidity in the atmosphere and you have one thermometer that is very good when the temperature is above freezing, but doesn't work otherwise, and you have another thermometer for when it's below freezing. Naïvely you would expect that these two devices would have different measurement errors, and even that their measurement errors change as you approach the edge of their measurement range, but if you didn't indicate this in your analysis, OLS would treat their data as being equivalently errorful.

As a result, it is generically prudent to examine the residual plot, that is, r_i as a function of x_i (and sometimes as a function of y_i!). Also, you should consider the distribution of r_i to ensure that it is at least somewhat normally-distributed (although we haven't yet talked about how to verify this quantitatively). Again OLS doesn't assume Gaussianity but the MLEs of the β's do. So if you are using the MLE solution to OLS then checking the assumptions of the two approaches seems wise! It is this philosophy that we wish to convey throughout this course: running modern machine-learning and statistical algorithms on your computers isn't hard. What is hard, and rarely done, are directed explorations of whether the approaches are valid and what your confidence in the results is.

As an example of such diagnostic plots, consider Figure 5. These plots can be immensely useful in determining whether there are any biases in your fits or any violations of the OLS assumptions. Also potentially useful is Seaborn's `residplot`, which plots r_i vs x_i with a special kind of smoothing line over the top, as shown here. For now, we leave this recommendation as a qualitative assessment that you can perform after fitting your model, but in the next sections, we'll establish how you can make this assessment *quantitative*. Finally, note that all fitting routines will generate residuals, so you can always make these figures and assess the appropriateness of the fitting assumptions.

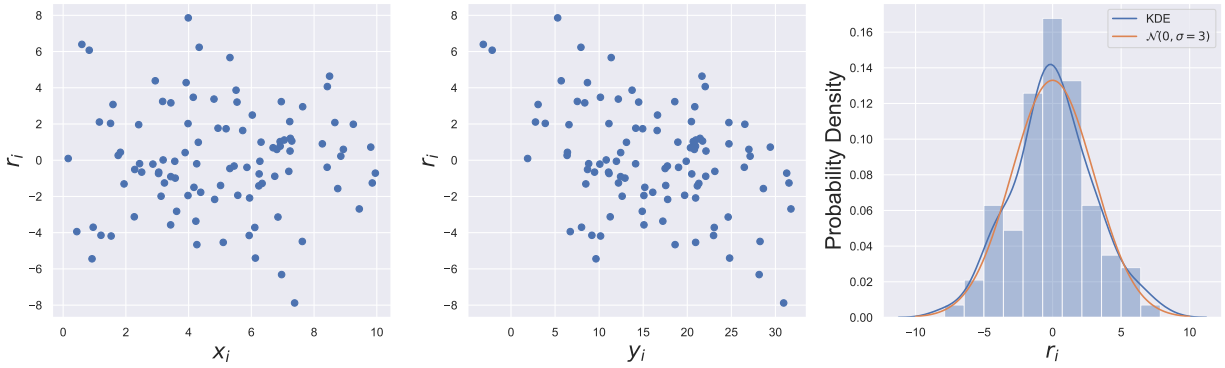


Figure 5: Three different ways of examining the residuals of an OLS linear regression fit. The left panel shows how r_i depends on x_i, if there were a violation of homoscedasticity, it would show up here as a relationship between r_i and x_i. The center plot shows r_i vs y_i. The downward trend in the residuals suggests that the model is overshooting the data when the response is small (large positive residual) and undershooting when the response is large (large negative residual). The right plot shows the distribution of residuals along side a normal distribution with zero mean and $\sigma = 3$, which is the value that was used to generate the data.

Try It Yourself

Using your data and model from Worksheet 2.2, recreate Figure 5. Discuss any patterns that you observe.

2.1.3 Maximum *a Posteriori* Formulation

We can also pose the problem of estimating β in a Bayesian sense. This section is more mathematically technical, so again, try and focus on the words and results if you are uneasy about algebra and mathematical notation.

In this formulation of the problem, we will again assume that the response depends linearly on the covariates plus some Gaussian noise so that we can write the likelihood as in equation 22. However, now we will explicitly model β_0, β_1, and σ (the parameter that OLS linear regression somewhat ignores) so that we can construct a posterior distribution for *each* of them.

This can be done several ways, but the algebra can be facilitated by choosing **conjugate** prior so that the math is easier to work with. Also, for ease of notation, we will be using $\vec{\beta}, X$, and $\vec{y}$ as defined in equation 25. The likelihood function is a normal distribution for $\vec{\beta}$, so a conjugate prior for $\vec{\beta}$ is also normal, while the likelihood function is a gamma dis-

tribution for σ, and so it turns out that an inverse gamma distribution provides a conjugate prior.

It's worth noting that you don't need to do this to construct a posterior - we can choose any prior and make the calculation empirically - but as your problem becomes more complicated, it can help to have algebraic forms for certain things. In any case, if you want to skip the rest of this derivation and use the result, that's definitely acceptable, but this reference should provide a workflow for how to attack new problems.

We write the priors explicitly as:

$$P\left(\vec{\beta}\middle|\,\sigma\right) = N_2\left(\vec{\beta}\,\middle|\,\vec{\mu}_\beta, \sigma^2\delta_\beta^2\mathbb{I}_2\right) \tag{27}$$

and

$$P(\sigma) = \mathcal{IG}\left(a_0, b_0\right). \tag{28}$$

Here, $\mathcal{N}_P$ is a multivariate Gaussian distribution with

$$\vec{\mu}_\beta = \begin{bmatrix} \mu_{\beta,1} \\ \mu_{\beta,2} \\ \vdots \\ \mu_{\beta,P} \end{bmatrix} \qquad \text{and} \qquad \Sigma_\beta = \sigma^2\delta_\beta^2 \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}.$$

($\mathbb{I}_P$ is the $P \times P$ **identity matrix**.) In our case, $P = 2$.

Notice then that these priors *each* introduce new parameters, $\vec{\mu}_\beta$, δ_β, a_0, and b_0, however, as we know, the effect of priors quickly fades with the addition of data, so the exact values of these parameters isn't that important. However, if we *do* have some information about $\vec{\beta}$ or σ, this method allows us to incorporate this information *by design*.

When using the Bayesian linear regression, it is often recommended to use $\vec{\mu}_\beta = \vec{\beta}_{OLS}$, the result from equation 26, and δ_β large (center the prior distribution on the OLS solution but give it a large spread to allow for variation). Similarly, a flat (uninformed) prior for σ can be set by using $a_0 = b_0 = 0.001$ (or some other small number, verify this yourself!).

It's also worth noting that we have set the prior for $\vec{\beta}$ as a *conditional distribution* on our other parameter σ. This is intentional, but not necessary in general for other problem, but does give the posterior some nice properties in this case that are not worth discussing here. You can see that Bayes Theorem is flexible enough to allow this when we write

$$P\left(\vec{\beta}, \sigma\middle|\, X, \vec{y}\right) \propto P\left(\vec{y}\middle|X, \vec{\beta}, \sigma\right) P\left(\vec{\beta}, \sigma\right),$$

where $P\left(\vec{\beta},\sigma\right)$ is the **joint distribution** for $\vec{\beta}$ and σ, and our properties of probability distributions tell us that $P(A,B) = P(A|B)P(B) = P(B|A)P(A)$ so that we can use $P\left(\vec{\beta},\sigma\right) = P\left(\vec{\beta}\middle|\,\sigma\right)P(\sigma)$ as our prior.

Finally then, we can multiply these distributions and normalize to find the joint distribution $P\left(\vec{\beta},\sigma\right)$. This will initially look like a big mess, but we're only interested in posteriors for just $\vec{\beta}$ or just σ, so we can **marginalize** (sum over σ to get $P\left(\vec{\beta}|\cdots\right)$ or sum over $\vec{\beta}$ to get $P(\sigma|\cdots)$ to get the following:

$$P\left(\vec{\beta}\,\middle|\,X,\vec{y},\sigma\right) = \mathcal{N}_P\left(\vec{\eta}_\beta,\Delta_\beta\right), \tag{29}$$

where

$$\Delta_\beta = \sigma^2\left(X^TX + \delta_\beta^{-2}\mathbb{I}_P\right)^{-1} \qquad \text{and} \qquad \vec{\eta}_\beta = \frac{\Delta_\beta}{\sigma^2}\left(X^T\vec{y} + \frac{\vec{\mu}_\beta}{\delta_\beta^2}\right).$$

The posterior for σ becomes

$$P\left(\sigma^2\,\middle|X,\vec{y},\vec{\beta}\right) = \mathcal{IG}\left(a_\sigma,b_\sigma\right), \tag{30}$$

where

$$a_\sigma = \frac{N}{2}+\frac{P}{2}+a_0 \qquad \text{and} \qquad b_\sigma = \frac{1}{2}\left[\left(\vec{y}-X\vec{\beta}\right)^T\left(\vec{y}-X\vec{\beta}\right) + \frac{1}{\delta_\beta^2}\left(\vec{\beta}-\vec{\mu}_\beta\right)^T\left(\vec{\beta}-\vec{\mu}_\beta\right) + 2b_0\right].$$

We can then find the maxima of these posteriors, or their moments, or their percentiles, etc. We can also draw samples from these distributions to empirically find moments, credible intervals, etc. if we don't want to do any more theory!

Finally, it's worth noting that when δ_β is large, then $\vec{\eta}_\beta \to (X^TX)^{-1}X^T\vec{y}$, the OLS solution. Also, the mean of an inverse gamma distribution is given by $b/(a-1)$, so again if δ_β is large and a_0 and b_0 are small, then the mean becomes

$$E[\sigma^2] \approx \frac{\left(\vec{y}-X\vec{\beta}\right)^T\left(\vec{y}-X\vec{\beta}\right)}{N+P} = \hat{\sigma}^2_{OLS},$$

which is the OLS point estimator for the variance .

This work may have seemed very technical, and it was. We omitted a ton of algebra that we can share with anyone who is interested (a later version of these notes will put

the details somewhere), but the point remains that we can also solve this problem using a Bayesian framework. Furthermore, this framework explicitly allows for the incorporation of previous knowledge and directly gives us distributional descriptions of our quantities of interest. Although we won't discuss it here, the Bayesian methods can relatively straightforwardly be adapted to account for heteroscedasticity, different models for the noise, or other more complicated variable dependencies. Given this flexibility, the algebra can seem somewhat worth the effort!

Try It Yourself

Following the example at the end of the notes, implement Bayesian OLS Linear Regression to your data from Worksheet 2.2. How do the distributions for the regression coefficients and noise spread (σ) compare to the values you used to make the data?

2.2 Confidence Regions for Model Parameters

We've now discussed how to make point estimates of regression coefficients by maximizing the likelihood of the data and by considering posterior distributions of $\vec{\beta}$, but we haven't finished the estimation until we've generated confidence regions.

You won't be surprised to see that there are 3 approaches to determining regions of confidence in your estimates: ML estimation, examining posteriors, and bootstrapping. You also won't be surprised to see that ML estimation will involve a lot of formulas and math, often relying on assumptions of homoscedasticity and normality of the residuals. We will present those calculations below. However, as is a theme that repeats itself throughout this course, the bootstrapping approach to producing estimates for parameter bounds is practical, straightforward, and generic.

The Bayesian approach requires no additional work than what you have already done to make the (MAP) estimate! The whole point of the Bayesian approach is that it gives a distributional estimate of parameters, and hence you need only report some statistic or region related to the spread of the parameter in question.

In this section, we'll just show an example of the different methods. As discussed earlier, the MLE confidence interval will be a formula that can be derived theoretically, the MAP estimator most naturally receives a credible interval that can be generated from the posterior, and we can bootstrap our OLS estimate to approximate the population distributions for our parameters and get confidence intervals computationally.

That is, we'll explain the details of the MLE confidence interval, but the credible interval and bootstrap confidence intervals can be generated using the methods given earlier. We'll show more detailed work and code for these methods at the end of the notes.

2.2.1 MLE

The formula for the confidence interval for a regression coefficient is

$$\left[\hat{\beta}_j \pm t_{\alpha/2,N-2}\hat{s}_{\beta_j}\right], \tag{31}$$

where

$$\hat{s}_{\beta_j} = \sqrt{\frac{1}{N-2}\frac{\sum_{i=1}^{N} r_i^2}{\sum_{i=1}^{N}(x_{i,j}-\bar{x}_j)^2}}$$

is the standard error in the j^{th} regression coefficient. $t_{p,\nu}$ is the p^{th} percentile of the Student's t distribution with ν degrees of freedom (`scipy.stats.t.ppf(p, nu)`). This is a result of the fact that regression coefficients (under OLS assumptions) are t-distributed, which is probably not obvious to you. It's worth noting that ML confidence intervals are often written as `Estimate` $\pm$ `Percentile`$\times$`Standard Error`, but you shouldn't assume this form unless you know that you can. In the case of regression coefficients, the **standard error**, is given by the formula above for $\hat{s}_{\beta_j}$.

More generically, it can be shown that the variance in β_j is given by $\Sigma_{j,j}$ (the j^{th} diagonal element of Σ, where

$$\Sigma = \sigma^2(X^TX)^{-1}.$$

This matrix is called the **covariance matrix** for $\vec{\beta}$ and is relatively easy to compute (it's often returned by your linear regression function). So once you have this matrix, you can take the square root of the j^{th} diagonal to get the standard error in that coefficient. Then the confidence interval can be found by taking $t_{1-\alpha/2,N-2}$ plus/minus the OLS estimate $\hat{\beta}_j$.

For a non-linear model, this formula provides an *approximation* to the confidence interval, but without knowing more precisely how our parameters are distributed, we cannot derive more precise intervals generically. In this case, as stated above, you may want to turn to bootstrapping as it provides an empirical route to estimating the spread in parameter estimates.

Try It Yourself

Take a stab at Worksheet 2.3. Once that's done, you should now be able to tackle the first three problems of Assignment 2.

2.3 Model Fitting vs Prediction

One goal then of a statistical model is that it performs well on *new data* and not just the data on which it was built. To this point, everything we've discussed is related to **training** the model – estimating the model parameters from the data. Parallel to the process of model fitting, we also want to assess the model's *predictive ability*, in particular, its ability to predict responses on previously unobserved inputs, or **generalization**.

As a result of these dual goals, it is often recommended in machine learning texts that you should split your data into two sets: a **testing** set for assessing generalization, and a **training** set for fitting the model (estimating the model parameters). You might see right away then that there are two errors that we want to monitor: the error in matching the training data and the error in predicting the test data. We'll refer to these as the training and testing error, respectively.

Obviously, a very good model will have both small training and testing errors, but this is not always realized with actual data. Instead we can note that the average training error will be smaller than the average testing error, so the performance of a model can be assessed both in its ability to shrink its training error and its ability to reduce the gap between the testing and training error.

More intuitively, these two errors directly correspond to the problems of **overfitting** and **underfitting**. Underfitting results when the model is unable to to describe even the training data set. This often corresponds to a model that doesn't have enough flexibility (or is simply inappropriate) to fit the data. As an example, think about using a linear function to fit a sinusoid. Overfitting on the other hand results from data that has incorporated the shape of the training data too much and can't predict new data. This often corresponds to a model having too much flexibility, so that it is inferring more from the data than may be justified.

In this way, we can control whether a model is likely to over- or under-fit by altering its "capacity" or "flexibility". Defining this quantity formally is difficult, but you can think of it as a model's ability to fit a wide variety of functions. For example, if we extend our

linear model to include a quadratic term:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2,$$

it has more capacity than the linear model in Equation 16. We could then use OLS to find β_2 in addition to the other regression coefficients and fit nonlinear models. As such, we say that models with low capacity may struggle to fit even the training set, while high capacity models might become too conformed to the training set by assuming it has features that it does not.

An appropriate, but useless, statement is that a statistical model will generally perform best when its capacity is appropriate for the true complexity of the task and the amount of training data. This topic of capacity will come back when we address regularization-based approaches to statistical modeling in Module 4. Surprising as it may seem, these approaches will attempt to modulate model capacity and fit the model in a way that is more consistent with the data.

Try It Yourself

Split your data from Worksheet 2.2 into a testing and training set. Use the training set to fit the linear model and assess the **prediction error**, the discrepancy between the model prediction and the actual test data. Now swap the roles of the two sets, what can you say about your model's generalizability?

Let's get back to practicalities. You have a dataset and you want to train your model and assess its generalizabilty. If you have enough data, you split it into training and test sets so that you can monitor performance and fitting at the same time.

How do you know that you have enough data to do this? Empirically, we would say that such a data is large enough that we've reached a regime where bootstrapped estimates are insensitive to adding more data. That is, you could run the tests that you've been doing in the worksheets where you ramp up the number of samples that you include in your estimates. If your estimates keep changing significantly (whatever that may mean) when you add more data, then you don't have enough data to split into two sets.

For the sake of argument, let's say that you do have "enough" data to split into two sets, and if you don't you will just have to proceed with even more caution! How then, should you split your data? Ideally, the testing and training sets are drawn from a single common distribution, so you can naïvely guess that breaking your data along the lines of $x < x_0$ and $x > x_0$ will be dangerous, because there might be real differences in your data

between those two sets. In such a case, not only might your training error be large because you don't have a wide-enough range of data to fit your model, your testing error will also be large because the model will never have seen data from the other region. Instead, the best we can usually do is to select our sets *randomly* from our full data set. One way to do this quickly in Python is to use `np.random.shuffle(np.arange(N))` to generate a shuffled list of *indices*. Then you can, for example, take the first 10% of these indices to grab your testing set data.

While this previous point may have been somewhat obvious, it's a bit less obvious what proportions we should split the data. If we allocate too much data to the testing set, then we'll get a bad fit, but if we allocate too little, then it won't be clear whether the predictive performance was due to the data in the testing set or the model. This is especially tricky if we're comparing two methods or models! How then can we balance this? The answer is called **cross-validation**, which we'll now describe.

Try It Yourself

Modulate the size and characteristics of your testing and training sets. How do training and testing error change as you modulate the size of the testing set? What happens if you break your data into two sets by putting a threshold on your covariate so that all the data below the threshold is training data and the data above is testing data?

2.3.1 Cross Validation

As we explained earlier, least-squares is just one of many ways that we could choose to optimize our model's parameters. In least-squares, we decided that minimizing the discrepancy between our data and our model was the most important objective. This is by far the most common tack taken, but an interesting alternative to minimizing these discrepancies is to ask that our coefficients are those such that the model is most *robust* to new data. That is, maybe we're less concerned that the model match the current data really well, and instead that it should match new future data well.

Alternately, when the parameter of interest is not a model parameter, but an analysis or algorithmic parameter, this perspective can be useful. Such parameters are called **hyperparameters** and are common in many machine learning or analysis algorithms.

But how can we perform such a fitting? We don't have *new* data, we only have the data we've collected! The most common way around this is to use the technique of **cross-**

validation. This method builds on the concept of testing and training data partitions in such a way that addresses some of our questions from earlier.

In particular, cross-validation will address the concerns that our test or training sets contain outliers or the data are not evenly distributed between the sets along some covariates by going one step further than simply breaking the data into two sets. Cross-validation says that we should repeatedly break the data into two sets, each time containing different partitions of the data, and assess the accuracy across these repeated partitions. (You can think of this as bootstrapping the accuracy assessment!)

There are of course many ways to partition your data, but the most common are called k-Fold Cross-Validation and **Leave One Out Cross-Validation** (LOOCV). In k-fold CV, one breaks the data set into k equal sized partitions (folds) and sequentially uses each partition as the test set while using the $k-1$ other partitions as the training set. Often $k = 10$ or $k = 5$ is used. The extreme case when $k = N$, the size of the data set, is LOOCV. This is because we are using all of the data except one point (leaving one out!) and trying to predict that one point. There are reasons to use or not use LOOCV, but generally, the biggest concern will be how long it takes for the code to run.

The idea then is that you can use CV to fit a model by performing CV repeatedly as you change model parameters. So if I am performing regression, I can search the β_0 and β_1 space to find where some prediction accuracy (the sum of squared residuals or maximum absolute residual, for example) is maximized. Alternately, if you have data that you know belong in different groups and are trying to generate a model to predict what group a data point is in, then counting the number of correct predictions can be useful.

We'll discuss cross-validation again later when we talk about model *selection*, not just fitting, but we introduce it here because it is a form of parameter estimation. It's worth noting that CV does not naturally give intervals of confidence, but you could construct a meaningful range on where you think parameters should be set by considering your accuracy vs. parameter curves.

Try It Yourself

Apply 10-fold cross-validation to your data from Worksheet 2.2. For each fold, use bootstrapping to estimate the regression coefficients (point estimate and confidence interval). Do these distributions overlap significantly from fold to fold?

2.4 Comments on Model Fitting

Throughout this chapter we have tried to stay as high-level as possible, diving into details only to illustrate useful examples. However, there are some comments that remain that do not belong in the text above. These are given here.

2.4.1 Model Selection is For Later!

You may have noticed that we did not explore the very interesting question of "what model should I use?" This was intentional; model selection is just as difficult and unanswerable as the topics discussed here and we feel it deserves its own thoughts. You will also see that many of the concepts here will rear their heads in that discussion as well, so this chapter will be good to understand in anticipation of that.

2.4.2 Many Adjustments; For Later

When discussing least-squares, we pointed out that the method of minimizing residuals was just one of many ways to fit models, and indeed there are many. In particular, non-linear models provide their own difficulties, and we'll discuss them shortly.

In the context of regression, there have been many good and powerful adjustments to the basic regression scheme that improve the predictive accuracy of those models and also aid in the development of a **sparse** model - a model that has fewer covariates. We hope to discuss this later, but if you are interested, the most popular adjustment is known as **regularization**, of which the most popular algorithms are known as **LASSO**, **Ridge Regression**, and a synthesis of the two, **Elastic-Net Regression**. These names are provided for your reference if you want to investigate them. These methods are particularly useful when the number of covariates, P, becomes close to or larger than the number of observations, N, in which case any model will be underdetermined. Unlike OLS linear regression, these algorithms come with *hyperparameters* that need to be tuned; this is most often done with cross-validation.

2.4.3 Non-Linear Models

Finally, as alluded to several times, non-linear models provide OLS with some difficulty. Not only is the calculus of finding their optima not as well-posed as in the linear case, it's also not clear that we should expect homoscedasticity of the errors, since *by definition* of the model, we expect differently sized changes in response at different values of the covariates.

As a result, the suggestion is that if you have a non-linear model you should attempt to linearize it. That is, if you want to fit an exponential function, take the logarithm; if you think your model is proportional to a square root of your covariate, then you should fit the squared model with OLS linear regression. In symbols, it is worse to try and fit

$$y = e^{-\beta x} \qquad \text{and} \qquad y = \sqrt{\alpha x},$$

than it is to fit

$$\log y = -\beta x \qquad \text{and} \qquad y^2 = \alpha x.$$

Try It Yourself

Create simulated data according to the model

$$y = (x + \varepsilon)^2 + 2(x + \varepsilon) + 1,$$

where you can pick the shape of the noise term ε. Plot your data. Try fitting this model using your linear regression methods by supplying covariates $X = [1, x, x^2]$. Then try fitting the equivalent model

$$\sqrt{y} = (x + \varepsilon) + 1.$$

Examine the regression coefficients and residual distributions. What do you notice?

3 Garcia and Phillips: Real Data!

For the assignment that accompanies this text, there is a data set, which is a part of the data used in this paper by Garcia and Phillips. In this paper, Garcia and Phillips generated a **thermodynamic model** of a famous **gene regulatory network**: the network around the ***lac* operon**. This model was exceptional for a few reasons, but among the most interesting applications was that the model could be used to estimate some quantities that are generally very hard to measure: the actual *number* of certain proteins in a given cell and the *binding energies* of certain proteins to DNA.

To understand why this is interesting and how they made and fit their model, we'll give a little background on gene regulatory networks, the *lac* operon specifically, and ther-

modynamical models. Then we'll discuss the data and how they used it in the paper.

3.1 Biology: Gene Regulatory Networks

The **Central Dogma of Biology** is a framework for how information is propagated from DNA to proteins. The general principle is that DNA is **transcribed** into messenger RNA (mRNA), and that RNA is **translated** into proteins. It's shown as a cartoon in Figure 6.

Of course, reality is not so neat, and there are actually arrows between each of the three components (DNA, RNA, and proteins) in every possible direction. That is, not only can RNA molecules generate proteins via translation, but they can also interact with DNA to affect transcription. Similarly, proteins don't just run off to never come back to the nucleus of the cell, they often directly regulate transcription and translation. As a result, biologists refer to the set of interacting DNA, RNA, and proteins as a **genetic regulatory network** (GRN). This name more aptly describes the biology in which a *network* of agents *regulate* the expression of one or more *genes*.

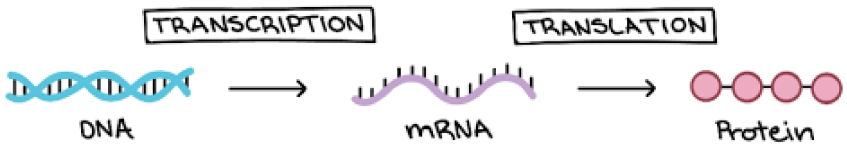


Figure 6: Illustration of the Central Dogma. Courtesy of Khan Academy

The *lac* operon GRN refers then to the set of proteins and chemicals that regulate the set of genes known as the *lac* operon. The *lac* operon, as shown in Figure 7, is a stretch of DNA in *E. coli* that consists of a promoter binding site, an RNA polymerase (RNAP) binding site, a repressor binding site called an *operator*, and three gene coding regions for the genes *lacZ*, *lacY*, and *lacA*. Based on the growth curves studied by Jacob and Monod, it was noted that *E. coli* bacteria, when placed in an environment with two food sources, glucose and lactose, first consume the glucose, then the lactose. This suggested that there is some sort of environmental control over when the bacteria would manufacture the enzymes needed for processing lactose.

This is indeed the case; the bacteria does not waste energy maintaining or creating enzymes for a food source when there is a much easier to consume resource, glucose, around.

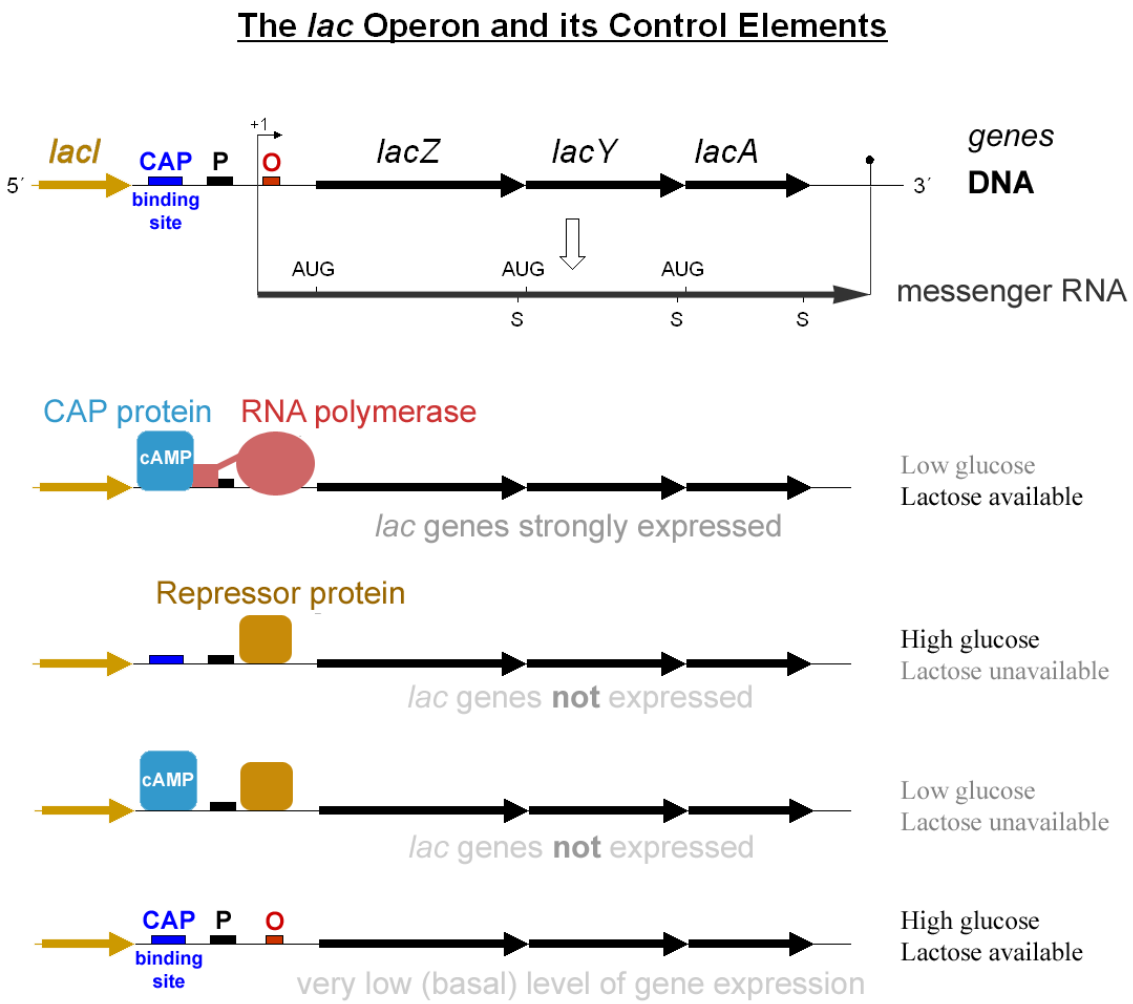


Figure 7: Diagram of the *lac* operon in each of its four states. Image courtesy of Wikipedia.

The effect becomes pronounced when only lactose is available, in which case many *lac* genes are expressed, or when there is a lot of glucose and no lactose, in which case almost *no lac* genes are transcribed.

This system has been extensively studied - it is probably one of the best understood regulatory networks we know of. It has been shown that when lactose is unavailable, a protein, known as a **repressor** binds to the DNA on the operator region, *physically blocking* RNAP, which transcribes the DNA into mRNA, from binding. Because this protein stops expression, it is called a repressor. On the other hand, when there is low glucose, another protein, called a **promoter** because it promotes the expression of the gene, binds to the DNA. This protein, which in this case is named CAP, makes it *easier* for RNAP to bind, thus increasing expression. In this way, the bacteria have direct mechanisms that control *transcription* and not just protein activity as was initially thought.

As a result of the work put into understanding this network as a simple model for all GRNs, this network is often taught in biology courses as an introduction to the topic. It is likely for a similar reason that Garcia and Phillips chose it to model in their paper. To understand why they needed a simple network to model, we'll first talk a bit about thermodynamics, whose principles they used to generate their model.

3.2 Thermodynamics

At this point it might not surprise you that thinking about things *probabilistically* is a profound and useful way of attacking problems. Indeed, we'll see that physicists have been using it successfully for centuries, and that it in fact underpins the entire discipline of thermodynamics (and quantum mechanics, but that's not relevant here), which is often called *statistical mechanics*. Thermodynamics is a branch of physics that attempts to codify the effects of temperature and heat into a set of principles.

One of the base ideas of thermodynamics is that while physically most things want to have a minimum energy, due to *thermal* fluctuations (random photons hitting atoms, molecules bouncing and jiggling), there is a non-zero likelihood that a system will have some non-minimal energy. Physicists have then shown that if you can then *enumerate* all the different states in which the system can exist and their energy levels, then you can predict the *probability* that the system will exist in any particular state.

Now, this might sound a bit ridiculous, if I have even a closed box of air, 1 meter to a side, I have on the order of 10^{25} molecules in that box, so enumerating all of the different ways that those molecules can be arrayed in my box should seem rather insurmountable.

More concretely, **Boltzmann's distribution** exactly describes the probability of observing any given state as a function of that state:

$$P(\text{state}_i) = \frac{e^{-\frac{E_i}{k_B T}}}{\sum_{\text{All States},j} e^{-\frac{E_j}{k_B T}}} = \frac{e^{-\frac{E_i}{k_B T}}}{Z}. \tag{32}$$

Z is known as the **partition** function, and is the part of the problem that is impossible: sum all the states and their energies.

However, it is often possible to write down the energies and number of ways that a system can be in a few *particular* states, in which case we can talk about the *relative probabilities* of those states by dividing the two Boltzmann distributions so that Z cancels out. This is often incredibly useful, as Garcia and Phillips note in the development of their model. We're not often concerned with the probability distribution across all possible states, but just across a few interesting ones. Known the relative likelihoods in such as situation is very powerful. In particular, we can see that

$$\frac{P(\text{state}_i)}{P(\text{state}_j)} = \frac{e^{-\frac{E_i}{k_B T}}}{e^{-\frac{E_j}{k_B T}}} = e^{-\frac{E_i - E_j}{k_B T}} = e^{-\frac{\Delta E_{ij}}{k_B T}}, \tag{33}$$

so that the relative probability of observing one state versus another is dependent on the *difference in their energies*.

If we consider our two states to be a molecule, say a protein, binding or not binding to another molecule, say DNA, then the relative likelihood of observing the molecules bound together compared to separate is directly connected to the binding energy of those molecules. In fact, this is often what we really mean when we talk about binding energies - we want to know the relative likelihood of the molecule being unbound versus bound.

In either case, this binding energy is a useful and interesting quantity to know, but is often difficult to measure directly. This is especially true in live biological systems, where getting a handle on such chemical measurements is confounded by the thousands of different processes occurring simultaneously. However, in posing a model that is explicitly based on thermodynamics, Garcia and Phillips were able to make *in vivo* estimates of the binding energy of the *lacI* repressor to DNA.

3.3 The Experiment and Image Processing

With this introduction, you are now ready to understand the model and measurements that Garcia and Phillips made. First, the goal of the paper was twofold: show that a thermodynamic model might describe gene regulation, and then that that model could be used to make estimates of quantities that are normally very hard to measure, like *in vivo* protein numbers.

This first point is interesting because it's not obvious that gene regulation should be an equilibrium process (as thermodynamics assumes), but it could be that it requires the use of *energy* (via ATP or something similar). Showing this model's explanatory power was evidence of how the underlying processes work.

Then they used their model to make a variety of predictions, most significantly of the repressor binding energy, $\Delta\varepsilon_{RD}$, and the repressor copy number, R. (The RD stands for **R**epressor-**D**NA to indicate the binding of the repressor to DNA.) They used other more complicated techniques and results to verify their model fits and found very good agreement. This suggests that such a model fit and set of experiments might be useful for future measurements of those quantities.

3.3.1 The Model

Specifically, the model they fit is given in Equation 5 of their paper:

$$\text{Fold Change in Expression} = \left(1 + \frac{2R}{N_{NS}} e^{-\beta \Delta\varepsilon_{RD}}\right)^{-1}, \tag{34}$$

where the *Fold Change in Expression* is the change in fluorescence in the presence of repressors compared to when there are no repressors in the system. R is the number of repressor proteins, $N_{NS} = 5 \times 10^6$ is the number of *non-specific* binding sites that the repressor could bind on the DNA that wouldn't impact expression, and $\beta = 1/k_B T$ is a Boltzmann factor.

3.3.2 The Data

The data consist of images of *E. coli* bacteria which have been modified so that the existence of certain proteins causes light at different wavelengths to be emitted. These images are known as **fluorescence** measurements, and are a common type of data in modern biological experiments. You will have access to several of these images for several mutant strains that are mentioned in the paper.

The images you have been given show the bacteria in three *channels*: a YFP channel, an mCherry channel, and a Phase-Contrast channel. These first two channels are named after the engineered fluorescent proteins that have been genetically inserted into the bacteria. The YFP (Yellow Fluorescent Protein) have been engineered to be attached to the *lac* operon, so that when those genes are expressed, the cell will fluoresce yellow. The mCherry (a red fluorescent protein) has been engineered to be expressed in all bacteria, hopefully enabling the cells to be seen compared to the background. The Phase-Contrast images are not channels corresponding to specific wavelengths, as in the previous channels, but instead look for *phase shifts* in the light to detect objects with different *optical densities*. As you can see in the images when you open them, this tends to highlight the *edges* of the bacteria, again potentially allowing us to extract them from the background.

Although this is not known in general, through a complicated calibration experiment, the authors were able to estimate the repressor copy number, R several strains and thus fit the model using fold-change in fluorescence as the response and R as the covariate.

3.3.3 Image Processing

In order to get the fluorescence (number of photons) of each bacteria in the images, one must do some **image processing**. That is, a picture to a computer is just an array of numbers. Each pixel location has a number telling the relative number of light energy that that part of the detector saw. The computer does not automatically know where the bacteria are. Furthermore, while we can look at the images and point to the bacteria, it is not possible to do this consistently when we have a lot of images. Thus, we must do some work to help the computer automatically find the bacteria in the images.

Specifically, we'll use the mCherry images to find where the *E. coli* are using two techniques called **thresholding** and **erosion and dilation**. The first method looks at the *distribution* of pixels across the image and tries to find a *threshold* where pixels that have more intensity than the threshold are bacteria and those that have less are not. This is a somewhat noisy operation, so to smooth out the detected regions, we use the **morphological operations** of erosion and dilation. These "erode" the detected regions to eliminate small noise, then "dilate" the remaining regions to restore their rough shape. This is shown in Figure 8. The last panel of this figure is titled "Size-Filtered Image" to indicate that there is also a filter applied that removes detected regions that are too small.

Once we have this last panel in Figure 8, we can use it to extract the only the YFP values from the YFP images that correspond to a bacterium and not the background. It's

worth noting that this is only *one method* for performing such processing, and in general the specific pipeline will vary from data set to data set. However, now that you've seen how this works, it may be worth considering this process when you design your experiment.

As a final note, in the pipeline code that I've given you, the YFP is extracted using processed mCherry images, not the phase-contrast images. This is purely practical - I found in my own experimentation that the mCherry gave more reliable results.

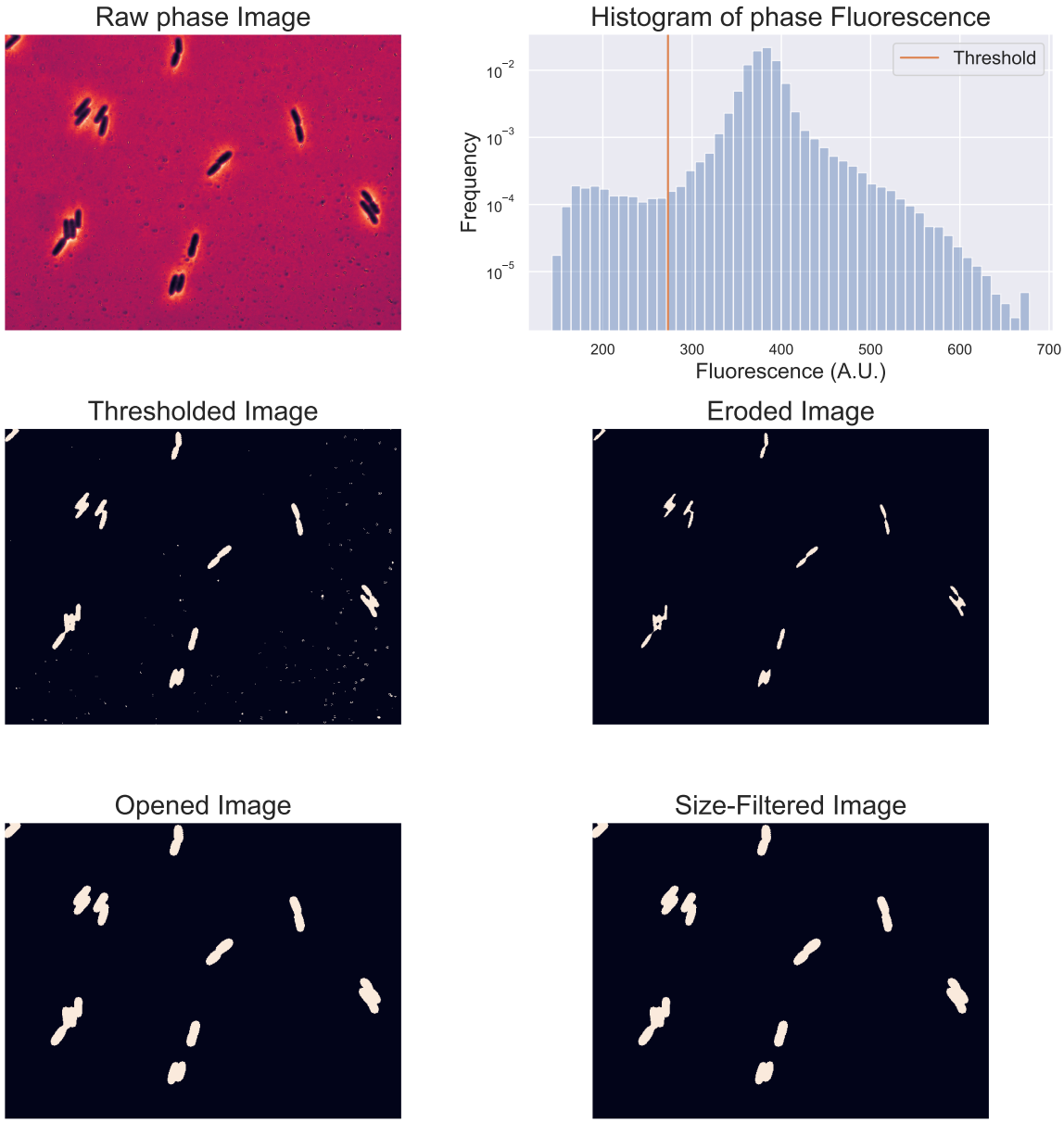


Figure 8: Example of simple image processing pipeline using a phase-contrast image. The top left panel shows the raw image. The top right panel shows the distribution of pixels in this image along with an automatically detected threshold. The middle left panel shows the binary image resulting from applying the threshold to the raw image. The middle right panel shows the result of applying an erosion operation and the bottom left panel shows the result of a subsequent dilation operation. The bottom right panel filters the detected regions for size, removing those that are too small; in this case, none were removed.

4 Review

So what was the point of this module? This module was meant to emphasize some of the most fundamental concepts in statistics and science.

First, begin quantitative isn't just reporting numbers; being quantitative requires having a fundamental understanding of what the number represents, what assumptions allow that number to be a representation of some part of your data, and how well you even know that number. We hope that you have started to see how being quantitative involves a deep grappling with uncertainty, and that there are many ways to estimate this uncertainty.

Second, we have introduced the way that you can confront data and models to each other. While we focused on the nuts and bolts of model fitting, actually doing science involves a back and forth between model generation and model assessment. Furthermore, we have presented model fitting as a specific type of parameter estimation so that the problems posed by model fitting are at the front of your mind when you are fitting your models as well. Specifically, we want you to ask the same questions when you're calculating a standard deviation that you do when you do linear regression:

- How likely am I to observe this estimate?
- How confident are you in this estimate?
- How much deviation from this estimate should I expect?

Within answering these questions, you have to assess questions of likelihoods, expectations, and prior knowledge. We point out that you can leverage your computer to answer these questions more concretely than theory.

Thirdly, we address the aspect of parameter fitting that extends beyond making models purely descriptive, but also *predictive*, by introducing cross-validation. This aspect of confronting models to data involves assessing the validity of fitting assumptions as well as whether your data are appropriate for the type of model you're considering. We demonstrate how computational techniques can be used to practically determine your model's tendencies to over- or under-fit.

Finally, we encourage you to take a step back from the details of the module, and see that unlike in Module 1, you are now beginning to use some of the basics to actually analyze data. In particular, we introduced you to important aspects of statistical thinking, problem solving and navigating quantitative problems by giving you exposure to the multiple approaches often used to analyze data, and new methods to calculate and simulate important

quantities.

4.1 Content Summary

- There are many ways to make **point estimates**
  - **MLE** seeks to maximize the **likelihood** function.
  - **MAP** seeks to maximimze the **posterior** function.
  - If you have the distributional form of the posterior, you can also calculate expected values, medians, or modes.
- Point estimates are only so useful, we need to have a region of **confidence**.
  - **Confidence intervals** measure the confidence to which future experiments will enclose the true parameter.
  - Confidence intervals require *theoretical* calculations
  - **Credible intervals** measure the region containing a certain percent of the *probability* of the parameter existing.
  - Credible intervals represent the *current* probability of θ having a specific value.
  - **Bootstrapping** allows for empirical calculation of MLEs and confidence intervals.
- Models have parameters. These parameters need to be estimated.
  - **Ordinary Least Squares** (OLS) is a framework that suggests that we should choose the parameters that minimize the sum of the squared residuals, $r_i = y_i - f(\vec{x}_i, \Theta)$.
  - When we assume the errors are normally distributed, the OLS framework can be posed as a maximum-likelihood problem.
  - When our model is linear ($y_i = \vec{x}_i\vec{\beta}$), then we can recover the MLE for $\vec{\beta}$ using equation 26. This is known as **linear regression**.
  - We can also use **Bayesian linear regression** to solve this problem with some relaxed constraints and informed priors.
  - We can also use bootstrapping to estimate parameters and their confidence intervals.
- **Cross-validation** is a method for systematically assessing the predictive power of your model by breaking your data into **training** and **test** sets.

  - Cross-validation is often used to fit **hyperparameters**, which are parameters of the algorithm or analysis, not of the model.
  - k-fold cross validation involves breaking the data into k sets and using each of the k partitions, or folds, as the test set in turn.

4.2 Learning Goals

In keeping with the overall learning goal structure of this course, the learning goals of this Module can be broken into the following specific actions.

By the end of the module, it is our goal that a student can...

- Manipulating and Visualizing Data
  - Illustrate estimates and regions of confidence
  - Illustrate model fits and uncertainty in those fits
  - Illustrate bootstrapped parameters
  - Illustrate and annotate plots of model residuals
- Performing Calculations and Simulations
  - Calculate MLEs and confidence intervals using mathematical formulas
  - Calculate MLEs and confidence intervals using *bootstrapping*
  - Calculate MAP estimates and credible intervals from computational posterior distributions
  - Perform OLS linear regression to get estimates of regression coefficients
  - Calculate estimates and confidence intervals of regression coefficients by combining bootstrapping and linear regression
  - Use bootstrapped or posterior distributions for parameter to assess the likelihood of deviations from expectation
  - Calculate the prediction error of a model
  - Implement cross-validation to calculate the prediction error
  - Generate estimates of fluorescence from images
- Thinking Statistically
  - Elaborate on the distributional nature of estimates
  - Discuss different methods of making estimates and their assumptions
  - Discuss different methods for making regions of confidence and their assumptions

  - Determine appropriate methods for estimating different quantities and their uncertainties
  - Discuss the assumptions and limitations of ordinary least-squares for fitting models
  - Interpret figures of model residuals to assess model performance

- Navigating Quantitative Problems
  - Critique different methods of making estimates and determine an appropriate method for a given data set
  - Critique different methods of making regions of confidence and determine an appropriate method for a given data set
  - Determine how and when it is appropriate to explore multiple methods for making estimates
  - Critique different methods for fitting simple models and determine and appropriate method among them
  - Assess the predictive ability and appropriateness of a model for a given dataset
  - Compare models using cross-validation

5 Details and Useful Code

In this section, we will outline some useful details and Python code examples that would have cluttered the main text of the notes. This section may be a bit more mathematical with less explanation than the rest of the notes, but we will still try and explain the main takeaways without relying on equations or algebra.

5.1 Derivation of ML Confidence Intervals for Exponential Rate Parameter

In this subsection we will detail the derivations of the formulas given in Problem 1 of Assignment 2 in which you are asked to calculate the confidence interval for an exponential rate parameter, λ in a few different ways. The work in this subsection is really only for students who are interested in the details and is not at all essential for a complete understanding of this module. As given in the Assignment, the useful takeaways are the formulas given in Equations 47 and 52.

In the Assignment problem, you consider two lists of $N = 1372$ time intervals, τ_i, which seem to be exponentially distributed based on their distributions (see Assignment 1 solutions for an example). Furthermore, we believe these intervals are exponentially distributed because it is known that the time intervals between Poisson-distributed events are exponentially distributed, and the best first guess at how bacteria run-and-tumble is that the switching is a Poisson process[1]. Based on these assumptions then, we write that the *likelihood* of observing any given τ_i can be written

$$P(\tau_i \,|\, \lambda) = \lambda e^{-\lambda \tau_i}, \tag{35}$$

where λ is called the **exponential rate parameter** (it has units of events/time; in this case, $[s^{-1}]$).

The likelihood then of all the τ_i can be written as

$$P(\tau_1, \tau_2, \ldots, \tau_N) = \prod_{i=1}^{N} \lambda e^{-\lambda \tau_i} = \lambda^N e^{-\lambda \sum_{i=1}^{N} \tau_i}. \tag{36}$$

We can find the maximum likelihood estimator (MLE), $\hat{\lambda}$ by taking the logarithm of both

[1] See Berg and Brown (1972) for experimental evidence or Berg and Purcell (1977) for a detailed discussion of the "Physics of chemoreception"

sides of 36 and setting the derivative with respect to λ to zero. We also use $\bar{\tau} = \sum \tau_i / N$ to derive:

$$\log\left[P\left(\tau_1, \tau_2, \ldots, \tau_N\right)\right] = N \log \lambda - \lambda N \bar{\tau} \tag{37}$$

$$\frac{\partial}{\partial \lambda}\left[\log\left[P\left(\tau_1, \tau_2, \ldots, \tau_N\right)\right]\right] = \frac{N}{\lambda} - N\bar{\tau} \tag{38}$$

$$\frac{\partial}{\partial \lambda}\left[\log[P]\right] = 0 \qquad \Rightarrow \qquad \lambda = \frac{1}{\bar{\tau}}, \tag{39}$$

as indicated in the assignment, so that we can use $\hat{\lambda} = 1/\bar{\tau}$ as our MLE for the rate parameter. However, this doesn't tell us anything about how $\hat{\lambda}$ is distributed, and therefore about our confidence interval for this estimate. To determine that we'll have to do a little work.

MLE of Rate Parameter

The MLE for the rate parameter of an exponential distribution is given by

$$\hat{\lambda} = \frac{1}{\sum_{i=1}^{N} \tau_i} \tag{40}$$

Our first method proposed in 1.a.i. of the assignment is obviously wrong, but we will make use of its logic to get going on the approximate method given in 1.a.ii [2]. In particular, the logic of 1.a.i. says that for a confidence level of α, the confidence interval of a normally-distributed quantity, X, (like the mean of i.i.d. random variables) is given by $[L, U]$, where we define L and U as

$$1 - \alpha = P\left(L \leq X \leq U\right) = P\left(\mu + z_{\alpha/2}\sigma \leq X \leq \mu + z_{1-\alpha/2}\sigma\right), \tag{41}$$

where z_p is the p^{th} percentile of the standard normal distribution, μ is the mean of X's distribution, and σ is its standard deviation. We can also write this as

$$1 - \alpha = P\left(z_{\alpha/2} \leq \frac{X - \mu}{\sigma} \leq z_{1-\alpha/2}\right). \tag{42}$$

If we were actually considering a mean, $\bar{\tau}$, then the CLT tells us that $\sigma = \sigma_\tau / \sqrt{N}$ so that we have

$$1 - \alpha = P\left(z_{\alpha/2} \leq \frac{\bar{\tau} - \mu}{\sigma_\tau / \sqrt{N}} \leq z_{1-\alpha/2}\right). \tag{43}$$

In the case of an exponential distribution, we have $\mu = 1/\lambda$ and $\text{Var}[\tau] = 1/\lambda^2$ so that this

[2] For more details, consider the post here.

equation becomes

$$1-\alpha = P\left(z_{\alpha/2} \leq \sqrt{N}\frac{1/\hat{\lambda}-1/\lambda}{1/\lambda} \leq z_{1-\alpha/2}\right) \tag{44}$$

$$= P\left(\frac{z_{\alpha/2}}{\sqrt{N}} \leq \frac{\lambda}{\hat{\lambda}} - 1 \leq \frac{z_{1-\alpha/2}}{\sqrt{N}}\right) \tag{45}$$

$$= P\left(\hat{\lambda}\left(1+\frac{z_{\alpha/2}}{\sqrt{N}}\right) \leq \lambda \leq \hat{\lambda}\left(1+\frac{z_{1-\alpha/2}}{\sqrt{N}}\right)\right). \tag{46}$$

This is the formula given in the assignment. Of course, this formula rests on the CLT and therefore only applies when N is "big enough" – it's approximate. To make it exact will take a bit more work.

Approximate Confidence Interval for Rate Parameter

An approximation to the confidence interval for an exponential rate parameter is

$$\left[\hat{\lambda}\left(1+\frac{z_{\alpha/2}}{\sqrt{N}}\right), \quad \hat{\lambda}\left(1+\frac{z_{1-\alpha/2}}{\sqrt{N}}\right)\right] \tag{47}$$

As noted here, coming up with a formula for the exact confidence interval will require that we make the somewhat odd observation that if a random variable X is exponentially distributed with some rate λ, then the random variable $Y = 2\lambda X$ is distributed like a χ^2 distribution with 2 degrees of freedom (this is what is noted in problem 1.b.iv.). More formally, if X is distributed according to $f(x)$ and $y = Y(x)$, then the distribution for Y can be found as

$$g(y) = f(x(y))\frac{\partial x}{\partial y}.$$

In our case, $f(x) = \lambda e^{-\lambda x}$, $y(x) = 2\lambda x \Rightarrow x(y) = Y/2\lambda$, and $\frac{\partial x}{\partial y} = 1/2\lambda$. Putting all this together gives

$$g(y) = \lambda e^{-\lambda\left(\frac{Y}{2\lambda}\right)}\frac{1}{2\lambda} = \frac{1}{2}e^{-\frac{Y}{2}},$$

which is the formula for a χ^2 distribution with 2 degrees of freedom (denoted χ^2_2).

This might have seemed esoteric, but knowing this lets us write the following interval for Y (and therefore X):

$$1-\alpha = P\left(\chi^2_2(\alpha/2) \leq Y \leq \chi^2_2(1-\alpha/2)\right), \tag{48}$$

where $\chi^2_\nu(p)$ is the p^{th} percentile of a χ^2 distribution with ν degrees of freedom. We're almost

there, now let's set $X = \tau$, our time interval, and note that if each $Y_i \sim \chi^2_2$ then the sum of $Y_i = N\bar{Y}$ is distributed according to χ^2_{2N} (this is can be shown for χ^2-distributed variables) so that we can re-write the above equation as

$$1 - \alpha = P\left(\chi^2_{2N}(\alpha/2) \leq N\bar{Y} \leq \chi^2_{2N}(1-\alpha/2)\right) \tag{49}$$

$$= P\left(\chi^2_{2N}(\alpha/2) \leq 2\lambda N\bar{\tau} \leq \chi^2_{2N}(1-\alpha/2)\right) \tag{50}$$

$$= P\left(\frac{\chi^2_{2N}(\alpha/2)}{2N\bar{\tau}} \leq \lambda \leq \frac{\chi^2_{2N}(1-\alpha/2)}{2N\bar{\tau}}\right), \tag{51}$$

which is the formula in the Assignment.

(Exact) Confidence Interval for Rate Parameter

An approximation to the confidence interval for an exponential rate parameter is

$$\left[\frac{\chi^2_{2N}(\alpha/2)}{2N\bar{\tau}}, \quad \frac{\chi^2_{2N}(1-\alpha/2)}{2N\bar{\tau}}\right] \tag{52}$$

As an example of the differences between these assumptions, consider Figure 9, where many estimates, $\hat{\lambda}$ were made using $N = 5$ and $N = 100$ randomly selected samples from our lists of intervals, τ^+ and τ^-. When $N = 5$ there is a significant discrepancy between the normal and χ^2 fits, with the χ^2 fit being much closer to the actual data than the normal distributions. When we increase the amount of data that goes into the sample to $N = 100$, the discrepancy between the χ^2 and normal distributions disappears, so that the fits have essentially the same shape.

5.2 Bayesian Estimation of Mean and Standard Deviation of Normally Distributed R.V.s

In this subsection we want to provide some code that will demonstrate how we can use Bayes' Theorem to estimate the mean *and standard deviation* of a set of normally distributed random variables. That is, we'll show the code behind Figures 1 and 2.

Recall that Bayes' Theorem consists of three important parts:

$$P(PARAM\,|\,DATA) \propto P(DATA\,|\,PARAM) \times P(PARAM), \tag{53}$$

where the left-hand side is the **posterior**, $P(DATA\,|\,PARAM)$ is the **likelihood**, and

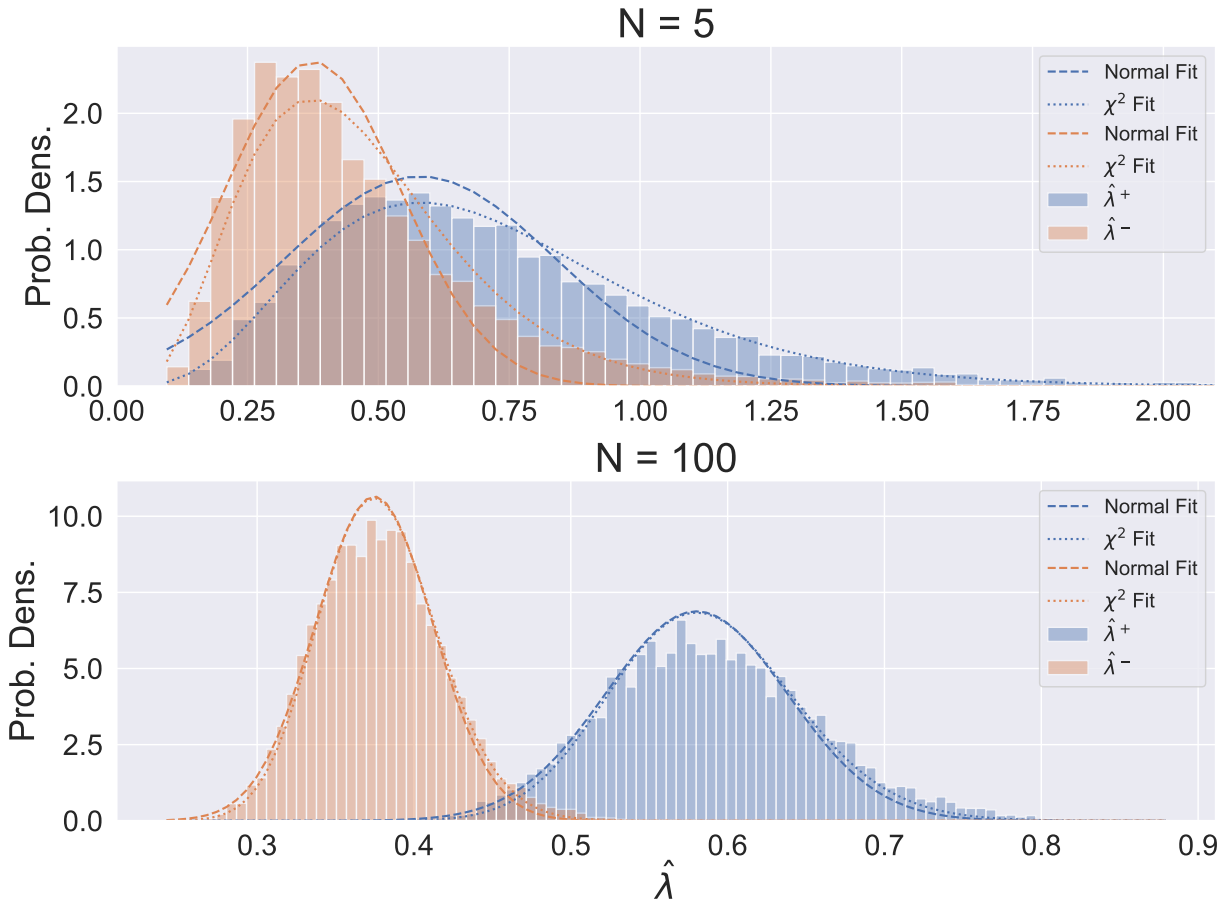


Figure 9: Distributions of bootstrapped estimates $\hat{\lambda}$ are shown. In the top panel, $N = 5$ samples are used in each bootstrapped estimate, and $N = 100$ are used in the bottom panel. Fits to normal and χ^2 distributions are shown as dashed and dotted lines, respectively, for each distribution, using the parameterizations described in Equations 47 and 52.

$P(PARAM)$ is the **prior**. The likelihood function will generally be similar to that used in a ML estimation, and the crucial difference that Bayes' affords us is the application of priors to create an *updating procedure*. That is, to generate the quantity we want, the posterior, we need to have a prior and likelihood and we need to multiply them together. However, for those of you encountering this formulation for the first time, actually executing this "multiplication" can be hard to wrap your head around. The key to keeping everything in order both in your head and in the code is to recall that your analysis is centered on estimating the *parameter*. This is different from your normal use of a likelihood function where you consider yourself *given* a parameter value and you input your data to get the numbers. In this formulation, we consider ourselves *given* the data and we will treat the parameter as our variable input.

In the code, this means that all of our operations and plots will be *functions of the parameter*. As a result, I like to recommend that the first step of implementing a Bayesian analysis is to establish a *grid* of parameter values over which your estimation will be performed (over which you'll calculate the prior, likelihood, and posterior distributions).

As an example, let's consider the abalones data set from Assignment 2. We're going to use the whole weight of the adult abalones as our data and we want to get an estimate for the mean and standard deviation. Code for loading this data is below:

```
from collections import Counter
import numpy as np
import pandas as pd
import scipy.stats as st

names = ['Sex', 'Length', 'Diameter', 'Height', 'Whole Weight', 'Shucked Weight',
         'Viscera Weight', 'Shell Weight', 'Rings']

abalones = pd.read_csv("abalones.csv", names=names)

weight = abalones['Whole Weight'][abalones['Sex'] != 'I'].values*200
```

As noted here, the continuous variables were scaled, so to return to units of grams, we multiply by 200. Once this has been done, we can determine that there are 2,835 samples in `weight`, ranging from 3.1g to 565.1g. Examining the distribution of `weight`, it seems unlikely that the mean is at the ends of this range, so let's set up a grid for the mean that goes from 100 to 400 grams. The following code creates a uniformly spaced grid of 1201 points from 100 to 400 (4 grid points per integer).

```
muGrid = np.linspace(100, 400, 1201)
```

Now that we have the values over which we will be implementing Bayes' Theorem, we need to generate the prior and likelihood distributions. We'll begin by creating a prior distribution.

First, let's consider a prior distribution for the mean, μ, that doesn't have any knowledge about μ, so it says that all possible values are equally likely (i.e. it is *flat*). We can calculate this by making a grid of ones the same size as `muGrid` and normalizing. Note that normalizing means that the area under the curve is 1, so that if we think of our function as being 1201 rectangles with width $\Delta x = 0.25$, then we need to set the height of those rectangles so that the total area is 1. In general, if you have a uniformly-spaced grid, you can normalize quickly by dividing by the sum of the y-coordinates and then dividing again by Δx, your grid spacing.

```
flatPrior = np.ones_like(muGrid)
flatPrior = flatPrior / (np.sum(flatPrior) * np.mean(np.diff(muGrid)))
```

This array now approximates the uniform probability density at each point in `muGrid`. To make sure that you have properly normalized, you can check that `np.sum(flatPrior) * dx` $= 1$.

Normalization

Note: In other places, we have advocated for normalizing by simply dividing by the y-coordinates without considering Δx. This is also ok, but is subtly different than the code above as it is not representing probability *density*, but a cumulative probability of being in the bin $[x-\Delta x/2, x+\Delta x/2]$. Since all computational distributions are discrete, the effect of this is that these quantities differ by the constant factor Δx (assuming a uniformly spaced grid), which will be important depending on what calculations you want to do with the arrays. In particular, if you are calculating moments, you need probabilities and not densities, so `flatPrior` would need to be multiplied by $\Delta x = 0.25$ for those calculations.

As a comparison, let's also consider a prior based on a (fictional) study that suggests that here

5.3 Conjugate Priors and Post of Exp Rate

Problem 1 of the assignment with code?

5.4 Building your own OLS Linear Regression

Implement the XTXXy formula.

5.5 Code for Bayesian OLS Linear Regression

Show what the calculation looks like in Python

5.6 Error bars on predictions

Code for different ways of showing error

Show error in *predictions*

Chapter Four: Empirical Hypothesis Testing

Eric Johnson and Madhav Mani

May 11, 2020

As you may have noticed in the previous two chapters, we have spent a good amount of time talking about how to think about, summarize, and extract distributional information from data, but we have not seriously discussed what to *do* with this information. This chapter will be our first foray into doing just that. In this chapter we will discuss the tools that a modern quantitative thinker needs to start to *test the validity* of a **hypothesis**.

We'll talk more about what this means shortly, but as an overview, this chapter will contain a discussion of:

- What it means to test a hypothesis,
- An introduction to several methods for comparing distributions,
- An introduction to some biological experiments and methods that you will need to complete the accompanying assignment.

There are several methods and perspectives presented here, but we would like to continue to advocate for *practicality*; when you are not sure what to do, simple simulations and bootstrapping can usually lead you in the right direction. When you have time, we recommend trying every method you know, and checking if they agree or disagree. Especially when using a theoretical result, use simulation to verify that the theory checks out. At the end of the day, it's better to be correct than dogmatic, and every statistical/computational/theoretical method can be tricked with the right data set.

1 Falsifying Hypotheses

As we suggested in the previous chapter, the goal of a quantitative analysis or a scientific experiment is generally not just to report estimates of parameter values. Generally, we want

to know more practically what we can *conclude* from these estimations.

Pretend, for example, that you are conducting a clinical trial on the efficacy of some drug in reducing inflammation in the joints of patients with arthritis. You have carefully double-blinded the experiment with placebos and a control group, and you have estimated the inflammation levels of all the groups. What do you do next?

Obviously, you can report the estimates (and intervals of "confidence"), but you didn't go to all of this work to just estimate these numbers; you wanted to determine the *effect* of a drug. That is, you *hypothesized* that this drug would do X, Y, and Z to reduce inflammation, and you want to assess how likely it is that this is true. This testing of hypotheses is the cornerstone of the scientific method and is exactly what we will discuss in this chapter.

More specifically, we know that it's at least useful to *think* about all quantities being distributional in nature, so we want to know more about how to *distinguish* distributions. In the drug study, asking whether the drug was effective is similar to asking whether the data sampled from the patients who received the drug **statistically different** from the data from the control patients. Put another way, we want to know the extent to which we can differentiate between the parent distributions for the level of joint inflammation in the two patient groups given the data that we've observed.

1.1 Null Distributions

Classically, this problem has been simplified into a decision between two outcomes: the **null hypothesis** and the **alternative hypothesis**. The idea is that the null hypothesis captures some large-scale negation of your scientific hypothesis. You think that the drug has an effect, the null hypothesis assumes there is no effect. Then, *given* the null hypothesis, we can ask the much more targeted question: "how likely is it to see my data given that the null hypothesis is true?"

This question is the crux of classical hypothesis testing, and the way to answer this question has traditionally been to know a lot of things about how your data are distributed, to make some assumptions, look up the correct statistical test, and finally use a table at the back of a reference book to estimate this likelihood. The idea is that theoretically, we sometimes know things about different quantities that show up frequently and we can write down formulas for how they are distributed. By setting a baseline using the "null" framework, we can use these theoretical results to test the likelihood of agreement with our data.

We'll briefly discuss this theoretical approach, but with the goal of demonstrating that

what's happening under the hood is that the tests are all attempting to compare distributions via more or less convoluted methods. Given modern computers then, it is easy to make simulated, resampled, or bootstrapped distributions of any quantities we want, so methods that work with these *empirical* distributions will be infinitely more applicable than a suite of theoretical tests.

1.1.1 Theoretical Null Distributions

Going back to our clinical trial, then let's say that we have our estimate of joint inflammation in the $N_C = 20$ control patients, $\bar{x}_{CONTROL} = 9.31 = \bar{x}_C$, with a 95% confidence interval of $[7.35, 11.26]$. This was calculated by calculating the sample mean and standard error in the sample mean, using the regular formulas. Similarly, we get $\bar{x}_{TREATMENT} = 7.39 = \bar{x}_T$, with a confidence interval of $[6.41, 8.37]$ from the $N_T = 45$ treatment patients. The individual measurements are shown as a histogram in Figure 1.

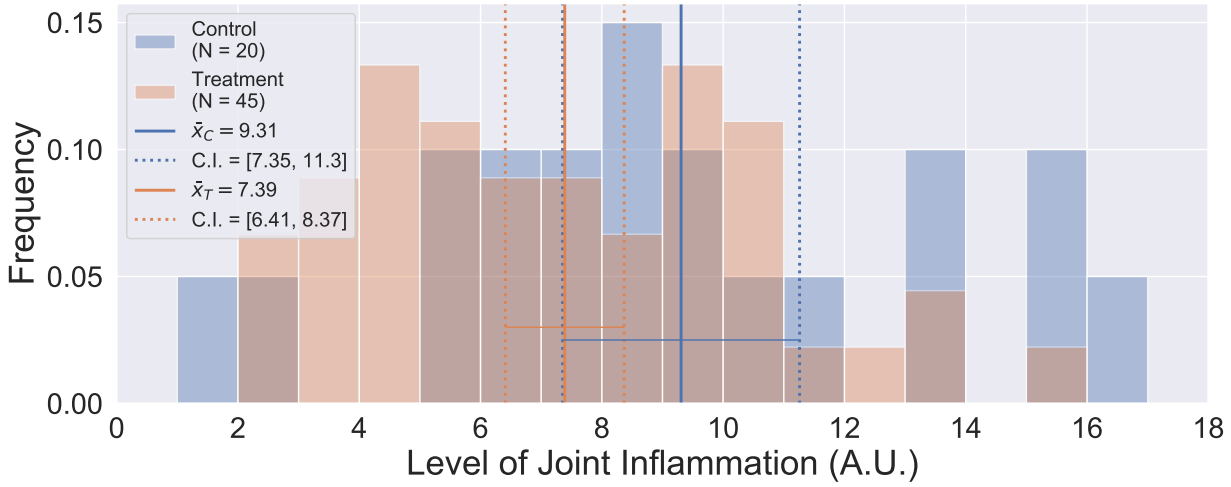


Figure 1: Frequency distribution of levels of joint inflammation in imaginary clinical trial for an arthritis drug. The control patients are shown in blue and the treatment patients are shown in orange.

We want to know then whether the drug has had an effect on the amount of inflammation in the patients. Based on Figure 1 you may be skeptical that there is an effect, since there is significant overlap between the distributions, but the means and confidence intervals suggest that there might be a difference.

This exact type of problem led to the development of **statistics**, or informative numbers, that were theoretically distributed in certain ways. The measurement of these statistics from data could be compared to theory and the likelihood of observing the statistic at that

value could be computed. That is, the classical approach is to find some *single quantity* based on our data that has some theoretical functional form, and then use that distribution to make conclusions. Depending on the statistic and the underlying data, the theory then changes, similarly to how the formula for the confidence interval of a parameter changed depending on the data, as we saw in the previous chapter.

At this point, we want to be clear, the classical approach is useful, but can be cumbersome and confusing. The empirical and computational approach advocated here is much more practical, and this will be emphasized in the assignment, worksheets, and the rest of the chapter. However, it's worth noting how the classical theory is basically working from the same perspective: we want to know the "difference" between a sampling distribution and some other distribution, whether it's another sampled distribution, a null distribution, or some other hypothesized distribution. As such, the rest of this subsection will show some of the details that make this differencing happen, but these details are not central to the main thrust of this chapter; you should make sure you understand the goal of the t-test, not the details.

Try It Yourself

Given this context, take a stab at Worksheet 3.1!

To be more concrete about our data, let's go through with the classical approach: we have two samples of a continuous variable, which we think are well described by a mean value. We then want to test the hypothesis that the treatment patients had a different (lower) inflammation level than the control patients. In symbols:

$$\begin{aligned} \mathcal{H}_1 : &\quad \mu_T \neq \mu_C \\ \mathcal{H}_0 : &\quad \mu_T = \mu_C. \end{aligned}$$

This is saying that $\mathcal{H}_0$, the null hypothesis (the hypothesis refuting our experimental hypothesis) is that there is **no effect** of the drug. Since this is an exploratory study, we don't really know the mechanism or scale of the drug's potential effects, so our alternative is that the drug has *some* effect so that the mean inflammation of the treated patient is *different* from that of the control group.

So how can we assess the likelihood of $\mathcal{H}_0$? Well, in 1908, a pseudonymous author

known as "Student"[1] noted that the distance between a sample mean, $\bar{x}$, and it's true value, μ, had a specific functional distribution.

Specifically, if

$$t = \frac{\bar{x} - \mu}{\hat{\sigma}/\sqrt{N}}, \tag{1}$$

where $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$ and $\hat{\sigma} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2$, then the **statistic**, t, would be distributed according to the function

$$P(t|\nu) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\Gamma\left(\frac{\nu}{2}\right)}\left(1 + \frac{t^2}{\nu}\right)^{-\frac{\nu+1}{2}}, \tag{2}$$

where ν is the *degrees of freedom*. In the case of the sample mean vs. the true mean, $\nu = N - 1$.

R. A. Fisher[2] later noted that this statistic and distribution could form the basis for a *statistical test*. In this test, μ could be set to μ_0, the mean of the null hypothesis, and the likelihood of getting a certain t-statistic (value) from the data could be explicitly calculated. This is now familiarly known as the **one-sample t-test**, because it uses a t-statistic with only one data set.

Of course, in our data, we have *two* samples: the control and treatment groups, so what do we do? The simplest thing (this is not necessarily the correct choice) is to say that our control group describes the null hypothesis, so we could use our one-sided t-test to assess the likelihood of observing our data from that distribution. That is, we have $\bar{x}_T = 7.39$, $\mu = \bar{x}_C = 9.31$, and $\hat{\sigma} = 3.26$, so that $t_{OBS} = -3.95$. Then $F(t;\nu) = \int_{-\infty}^{t'} P(t|\nu)\,dt'$ is the CDF of the t-distribution, so that $F^{-1}(q;\nu)$ is the *quantile* or *percentile* function. Then we can calculate that the likelihood of observing a value t **equal to or smaller than** t_{OBS} is $F(t_{OBS};\nu) = 0.014\%$.

We haven't specified that we're looking into the treatment group being *smaller* than the control group, so we should also ask for the likelihood that the means would be separated by a similar distance in the other direction. This gives $F(-t_{OBS};\nu) = 0.014\%$, so that the total likelihood of observing a mean that far away from our true value is $P(|t| \geq |t_{OBS}|) = 0.028\%$.

This probability is known as a **p-value**, and based on this probability, we either

[1] His real name was William Sealy Gosset; Student was his pen name. Also, while the test is named after him, the distribution was explicitly noted as far back as 1876 by Helmert and Lüroth, among others.

[2] Fisher popularized many of the methods in classical statistics, including MLE, hypothesis testing, and ANOVA. However, he was an avowed supporter of eugenics and segregation, so we should be careful not to tout him too highly.

accept or reject the null hypothesis. Classical hypothesis testing says that you *a priori* pick a significance level, say 5%, and you reject the null hypothesis if the p-value is below that level. The idea being that if the likelihood of observing the data from the null hypothesis is small enough, we are willing to accept the alternative hypothesis. As we'll talk about later, this formulation of a statistical test actually provides only *weak* evidence that your alternative model is correct; the test **falsifies** the null hypothesis, it doesn't *validate* the alternative.

In any case, you may still be looking at Figure 1 and thinking that we should not be so confident that these distributions are different from each other, and indeed, our one-sample test basically ignored all the information we have about the control group except the mean. Having noticed this, clever statisticians have come up with modifications to the t-test to cover cases when we are comparing two samples with equal variances, different variances, differing number of samples, interdependence between the samples, etc. Many of these modifications have been incorporated into modern programming languages, for example `scipy.stats.ttest_ind` reports that $t_{OBS} = -2.004$ so that $p = 4.94\%$. However, looking at the documentation, this method assumes that the variances are equal between the two populations, which we know isn't true. Turning the keyword `equal_var=False` now reports that $t_{OBS} = -1.822$ so that $p = 7.85\%$.

At this point, you may be feeling either relieved or more skeptical. On the one hand, incorporating more of our information about our control group resulted in much less confidence that the means of the distributions are different, but on the other hand, we've gotten three qualitatively different answers by changing functions and changing a keyword. Our first attempt said the means are extremely different, the second attempt put the difference right on the edge of 95% significance, while the third attempt moved us even further away from assessing that the treatment groups are different. It is partly because of this somewhat unpredictable sensitivity to the different tests' assumptions that we don't recommend building a quantitative toolbox around these methods.

Instead, what we'll stress throughout the rest of this chapter is that rather than worrying about the specific function that describes your null distribution or your test statistic, if you can empirically generate a sample from these distributions, you can apply a more general suite of tests that require less "institutional" knowledge. In the next subsection we'll discuss more precisely what this means both in the context of our drug trial example as well as for generic data sets.

1.1.2 Simulating Null Distributions

As noted above, and as will be come more apparent as we introduce more hypothesis testing methods, hypothesis testing generally boils down to expressing your null hypothesis as a distribution of some quantity of interest and then comparing your data's distribution to that null. As shown in the previous sub-section, this can mean noting that the difference between two means is t-distributed, but in general, this process does not need to rely on theory alone. Specifically, this section will illustrate that null distributions can often be generated *computationally* via simulation, bootstrapping, or somethings in between.

We'll consider generating null distributions via simulation more extensively later, but generally the principle is to try and replicate a process corresponding to your null hypothesis *in silico* (in your computer). In the example of our drug study, we could have simulated many trials of N_T patients by randomly drawing from a normal distribution with the same mean and variance as our control patients. From each of these simulated trials, we could then calculate our various t-statistics and create a simulated distribution of possible t values from data with such a mean and variance. We can then compare our t-values from the real data and ask how often we see such values in our simulations. This is done in the left panels of Figure 2 for each of our t-statistics: the naïve statistic as calculated in Equation 1, the statistic adjusted for the variance in the control group, and the statistic adjusted for the differing numbers in the two groups.

Alternately, we could have "bootstrapped" some new data sets from the control group by resampling from the control data N_T times with replacement. This generates a similar distribution of t-statistics which are shown in the right column of Figure 2.

In either case, we are using the control group as an estimate for the "no effect" distribution, and comparing our calculated statistics to that distribution. If we consider the eCDFs of these t-distributions, we can calculate an "empirical" p-value, which are shown in the bottom panels of Figure 2. We see that in both cases, we recover values that are consistent with the theoretical results reported earlier: the likelihood of observing the naïve t-statistic is very small, but the other two statistics are much more likely, to the point that they would not be sufficient to reject the null hypothesis at at a 95% significance level.

As a point of emphasis, the theoretical distribution for the t-statistic is overlaid in black in the top row of Figure 2. This should help convince you, albeit in a qualitative way, that we have not gone too far astray from our theoretical results with these simulations. In fact, we hope to convince you that this qualitative agreement holds up generally, so that simulation can be your *first* assessment of a hypothesis' validity.

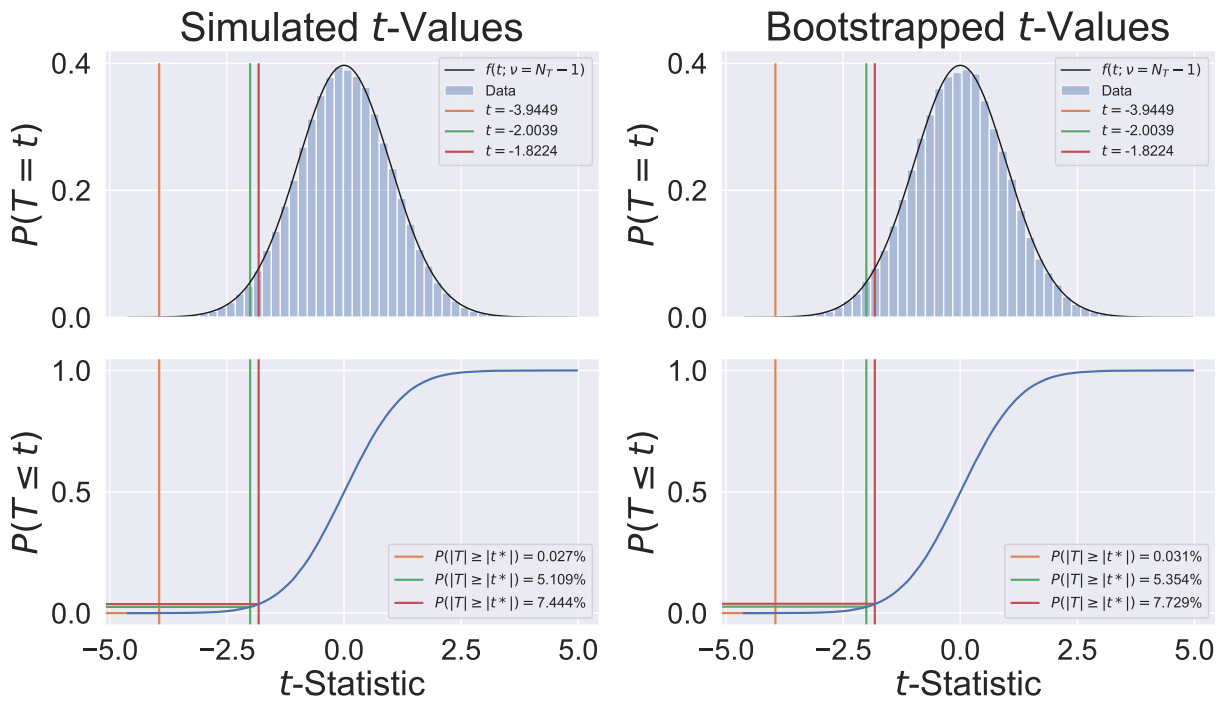


Figure 2: Distributions of t-statistics generated by (LEFT COLUMN) drawing $N_{RESAMP} = 100,000$ samples of size $N_T = 45$ from a normal distribution with $\mu = \bar{x}_C$ and $\sigma^2 = \hat{\sigma}_C^2$, and by (RIGHT COLUMN) drawing $N_{BOOT} = 100,000$ samples of size $N_T = 45$ with replacement from the control group data. The vertical lines indicate the location of the various t-statistics calculated from the actual treatment and control group data, as described in the text. The empirical likelihood of observing values less than these statistics, as calculated from the eCDF, are shown in the bottom panels. Not shown is that these values are calculated assuming a 2-sided test, that is, the reported numbers correspond to $P(|T| \geq |t^*|)$. This is omitted for illustrative purposes only.

Try It Yourself

With this information, you should now be able to tackle Problems 1-3 in Worksheet 3.2.

1.1.3 Simulating more complex null distributions

Of course, we're not always simply trying to compare the summary statistics from two groups of data, sometimes we have many groups of data, or we're trying to assess the existence of a relationship between two or more quantities. In such cases, we can extend our "bootstrapping" to each of our groups of data, covariates, or responses in a manner similar to that shown in Figure 2 by resampling our data from the empirical **marginal distributions** of each group / covariate / response.

Formally, the marginal distribution is the distribution generated from a joint distribution by integrating out the other variables in the joint distribution.

$$P(X) = \int \int P(X, Y, Z)\, dY\, dZ \tag{3}$$

In equation 3, $P(X)$ is the marginal distribution for the quantity X after Y and Z have been integrated out. Recall that an integral is a special type of summation so that you can also think of this discretely as

$$P(X = x_i) = \sum_{j=1}^{N_Y} \sum_{k=1}^{N_Z} P(x_i, y_j, z_k). \tag{4}$$

That is, the marginal is the distribution of X after averaging over all possible values of the other variables.

To be more concrete about how to do this resampling, consider a data set composed of N observations of two measurements, x_i and y_i, and we want to test whether there is a linear relationship between them. To do this, we want to generate a distribution of regression coefficients corresponding to the hypothesis that there is *no relationship* between X and Y, but how can we do this? It is relatively easy to generate many lists of numbers that have nothing in common (that have no *covariance*, to be precise), but just as with the drug trial we didn't want to compare our t-statistic to a distribution that had nothing to do with our control data, in the regression case, we want to preserve some information about the information we've gathered. This is accomplished by creating new fake datasets where

Figure 3: Example of generating a null data set by resampling from marginal distributions. The left panel is the original data, with the marginal distributions to the top and right. The right panel contains one resampled data set, with the marginals again shown on the top and right.

each measurement is resampled with replacement from the marginal distribution for that measurement.

This is perhaps more easily understood visually; consider Figure 3, where some X and Y data are shown in the main panel on the left, with the marginal distributions for X and Y shown on the top and right, respectively. We can think of the marginal distribution for X as the density of points in any X-location as we sum across the Y-axis. This can be thought of as the "shadow" of the joint distribution from the perspective of the X axis only. We can make a similar shadow for the Y data.

Notice in this data that while the scatter plot makes the data seem vaguely linear, the density of points in the X-direction is not uniform, but might instead be bimodal. Depending on how we fit our model, this bimodality may be more or less important, so we should seek to capture null distributions that incorporate this feature of the data while still removing the hypothesis that X and Y are related. To maintain this bimodality in a data set where X and Y are not related, we randomly sample N new X-values with replacement from our

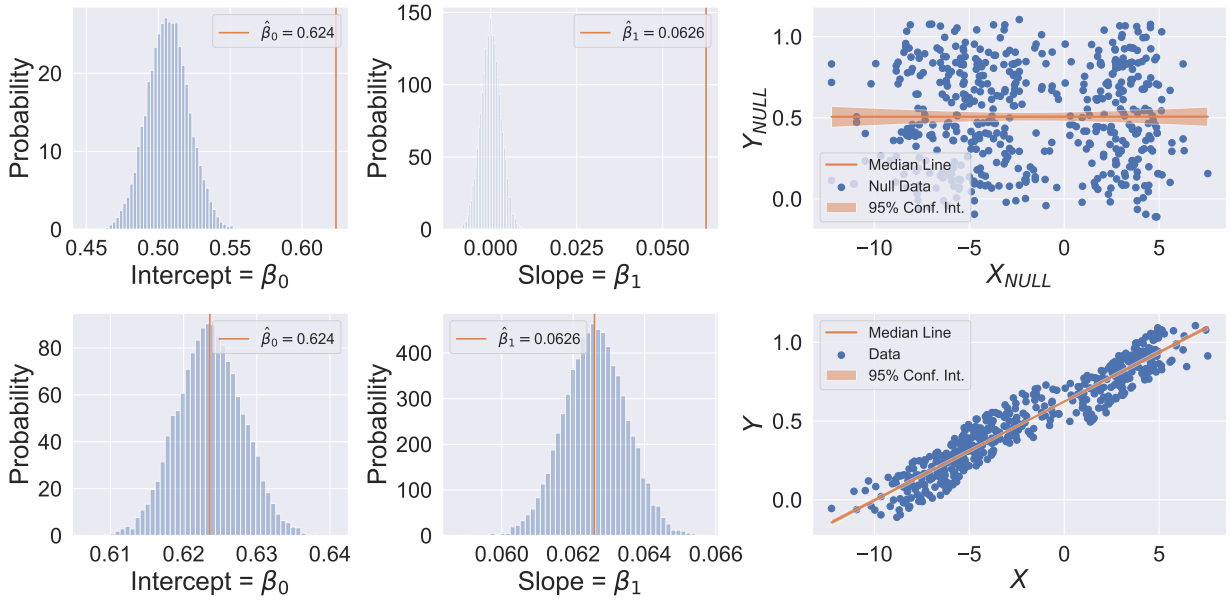


Figure 4: Example of null distribution for regression coefficients. (TOP LEFT, CENTER) Null distribution for linear model intercept and slope. Orange vertical lines indicates the OLS estimate from the original data. (TOP RIGHT) Scatter plot of single null data set with median and 95% confidence line plotted in orange. (BOTTOM LEFT, CENTER) Bootstrapped distributions for regression intercept and slope. Orange vertical lines indicates the OLS estimate from the original data. Scatter plot of original data with median and 95% confidence line plotted in orange.

data, *then* we sample N new Y-values (with replacement) from our data. We put these together as a "null" data set.

Such a null data set is shown in the right panel of Figure 3. Notice that the marginal distributions of X^* and Y^* are similarly shaped to those in the left panel; this is by design! We sampled values of X according to their relative likelihoods, so we should get a random sample with the same frequency of any X-value. However, we now see that there doesn't seem to be a relationship between X and Y. This is again what we hoped for; we drew the samples separately, so there's no way for there to be cross-information between the two.

We can then do this many times and fit a linear model to each of the null data sets to generate a null distribution for our regression coefficient. An illustration of this is shown in Figure 4, where many null data sets have been fit to generate the regression coefficients in the top row. The coefficients from the data are shown in the bottom row along side a bootstrapped distribution to show the spread in the estimate.

Try It Yourself

Try out Worksheet 3.3 to practice this yourself!

1.2 A Bayesian Perspective

In this section, we just want to make a few notes about the hypothesis test framework as presented thus far. In this framework, we have posited that we can falsify null hypotheses as weak evidence of the validity for an alternative hypothesis. This is not the only way to go about the problem of testing a hypothesis, an important alternative is the use of Bayesian posterior likelihood ratios. In this method, given two models, we can construct posterior distributions for our parameter or statistic of interest, and calculate the ratio of the distributions to assess where data is more supported by one hypothesis versus another.

Of course, Bayes Theorem allows us to write

$$P(MODEL|DATA) \propto P(DATA|MODEL)P(MODEL), \tag{5}$$

where now $MODEL$ may be meant to encompass a hypothesis, particular parameter values, and other details. In this way, if we could enumerate all the possible models and parameter combinations and hyperparameters, we could definitively determine the most likely hypothesis. This is obviously impossible, so restricting ourselves to comparing two models, or a few models, at a time seems more reasonable. In this way, the null-alternative approach is still

useful, and is generically the most practical approach.

Finally, going into the next section where we dive more directly into the methods of comparing distributions, we note again that Bayesian analyses return entire distributions and thus can be more naturally incorporated into some of these methods.

2 Comparison of Distributions

The general principle of this chapter is that we can test the validity of a hypothesis by quantifying the *null hypothesis* as a distribution of some quantity and comparing our data to that distribution. We can perform this comparison in many ways, as we'll outline below, but the underlying goal of each of them is to assess the **likelihood that our data are consistent with the null**.

For example, if our hypothesis is that an arthritis drug *had an effect*, then we can consider our control distribution as the null, and assess the likelihood that the control distribution could have generated the treatment data *by random chance*. Similarly, if we believe that there is a relationship between two quantities, X and Y, which we want to measure with linear regression, then we can consider the null distribution of regression coefficients found by fitting data generated from the joint distribution of the marginals, as explained in the previous section. We would then compare our regression coefficient (or distribution of coefficients) to this null distribution to assess the likelihood of observing our data's regression coefficient (distribution).

Throughout this chapter, we will attempt to illustrate the different ways that the distributions can be compared, in each case using the two previous examples (and sometimes other useful examples). However, we will sometimes use different characterizations of the null or data distributions. Sometimes we will use a bootstrapped distribution, sometimes a parametric resampling, and sometimes a Bayesian posterior distribution. These different ways of expressing the distributions should not be considered special to whatever distribution-distinguishing method we're using, but are only emphasizing that you have many ways of generating a null or data distribution and you should use whatever is easiest for you to understand.

2.1 Confidence and Credible Intervals

You may be concerned that we put a lot of effort into generating confidence and credible intervals just to ignore them going forward. This is not the case, we will will always ask that

estimates of parameters and quantities be contextualized with an interval of some sort, and this is always what one should report in any scientific conclusions. However, these intervals are *not* specifically useful in the context of distinguishing entire distributions because they (by design) ignore some of the parts of a distribution that might be useful for discerning differences. For example, if we have two distributions with similar means, but different confidence intervals, as shown in Figure 5, we can use the overlap in the confidence intervals to say that it's *possible* that the distributions are similar, but not necessarily how *likely* it is that they're different. That being said, if we have two confidence intervals that are *disjoint* (completely non-overlapping), then it does suggest that it is relatively unlikely that the distributions are the same.

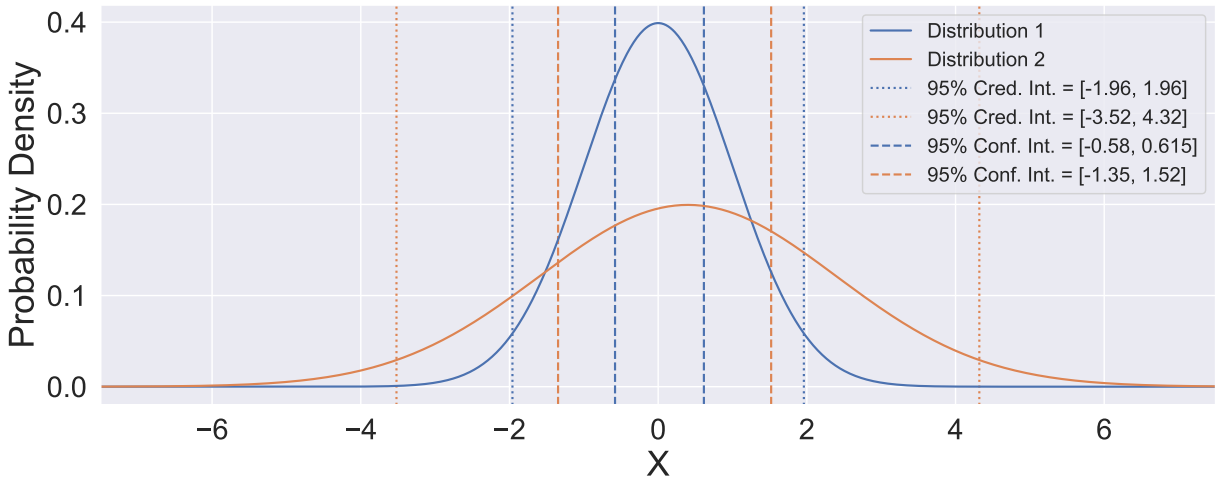


Figure 5: Example of two normal distributions with very similar means and different variances. The dashed lines indicate the confidence intervals for the mean generated by sampling 10 random variables from each distribution. The dotted lines indicate the $\alpha = 0.05$ credible region if these two curves were posterior distributions.

As examples, consider the drug trial data, where the confidence intervals in Figure 1 suggest that there is significant overlap in the distributions. However, our t-statistic calculations (which we take with a huge grain of salt) suggested that the likelihood of observing such a separation of means was $< 10\%$. This should serve to underscore that we do not have a general way to go from confidence interval overlap to likelihood that distributions are different. On the other hand, in the regression example shown in Figure 4 shows that the null and data confidence intervals are completely disjoint, confirming our intuition that there is a real relationship between the data.

On the other hand, credible intervals do contain some information about where we

expect the quantity to be distributed (not just our ability to capture the mean), so we can be a bit more confident in situations in which two intervals overlap, since we know that there is some probability density in those regions. However, in a case where we have distributional information, we'll see that we can do better than simply comparing intervals.

The takeaway here is that we can use non-overlapping intervals as qualitative evidence that distributions are different, but we need to use other tools to make this assessment quantitative. Furthermore, if confidence intervals overlap, it doesn't necessarily give us information about distributional "closeness."

2.2 p -Values

Before we get into methods that more directly compare distributions in a more global sense, it is worth discussing the standard method of comparing a single **statistic** to a null distribution for that statistic and then assessing the **cumulative probability** of seeing a statistic *with that value or smaller* (or larger), which we call a **p-value**. We performed this technique above when we talked about the t-test, but it's worth reinforcing that this is a more general technique that is applicable outside the suite of standard tests.

Specifically, the technique of calculating a p-value reflects the idea that we should seek to summarize our distribution or distributional-distance into one number called a statistic. We then consider how that quantity is distributed and ask whether it's reasonable to see such a deviation as we get from our data. Several of the following methods for comparing distributions will perform this consolidation and comparison, where the differences in methods will then be due to how the statistic is calculated, what the statistic is representing, and how the statistic is distributed.

It's worth noting however that you are not restricted to the class of statistics that you can find in a textbook. If you have some quantity of interest, say $1/(Var[X])^{x_2^3+|x_1|+2}$, you can treat that quantity as a statistic, generate a null distribution for the quantity, and use a CDF to calculate the likelihood of getting your data's result.

Calculating a p-Value

Given a data set, X, and a statistic $\theta = \theta(X)$, we can calculate the p-value for our statistic and data by following these steps:

- Computationally or theoretically, generate $f(\theta) = P(\Theta = \theta)$. If doing hypothesis testing, this should be the *null* distribution for θ.

- Computationally or theoretically, calculate the CDF, $F(\theta) = \int_{-\infty}^{\theta} f(\theta')d\theta'$
- Depending on what you want to test, your p-value,

$$p_\theta(X) = \begin{cases} F\left(\theta(X)\right) & \text{Left-Sided } p\text{-value} \\ 1 - F\left(\theta(X)\right) & \text{Right-Sided } p\text{-value} \\ F\left(\theta(X)\right) + \left[1 - F\left(\theta(X)\right)\right] & \text{Two-Sided } p\text{-value} \end{cases} \tag{6}$$

Now that we see how this works, let's reconsider our drug trial data. Instead of starting from the situation where we assume that we should use a Student's t-test because we were taught that in high school stats, let's approach this problem from a more generic perspective. In this case, considering Figure 1, we can see that it seems that both the mean and variance of the control and treatment populations' inflammation levels are different. However, we're hoping this drug will reduce the *average* patient's inflammation, so we want to assess the significance of seeing the treatment group's mean being less than the control group's mean by $\sim$2. That is, our first take on a statistic is

$$\theta(X) = \bar{X} - \bar{X}_C, \tag{7}$$

where $\bar{X} = 1/N \sum_{i=1}^{N} x_i$. In the case of our data, we get $\theta = -1.92$.

Then the null hypothesis in this case is that there is no actual difference in the means between the two groups, and our observed deviation is simply due to random sampling. To test this, we can bootstrap (sample with replacement) our control data to get samples of size $N_T = 45$, and we can calculate the null distribution for our statistic, θ. This is shown in Figure blah.

Of course, we can see from Figure 1 that bootstrapping the control data will never give us some of the observed values of the treatment group, so we could also simulate the null distribution by bootstrapping the *treatment* group and asking how often we see no difference in the means. This is shown in Figure blah.

Interestingly, these methods don't yield the same result, as we can see in Figure blah, the distribution of θ generated by bootstrapping the control group says that $P(\theta \leq -1.92) \approx 5 \times 10^{-4}$, but bootstrapping the treatment group says that $P(\theta \geq 0) \approx 4 \times 10^{-5}$. Why might this be? Referring to Figure 1, we can see that the two groups seem to have different spreads, with the control group having a wider range of data than the treatment group, even with half the observations.

To address this then, we could update our statistic to incorporate some information about the variances of our groups. We would probably also want to introduce some dependence in our statistic on the number of data points in each group. This is exactly what statisticians have done with the modifications to the t-statistic.

Now, this is all perfectly legitimate to do, but you should realize that as your statistic becomes more complicated, it becomes more difficult to explicitly nail down the null hypothesis. For example, if we assume that the two groups have the same variance, which variance should we use? The control group's, the treatment group's or some combination of the two? What if we don't want to assume they have the same variance, but we want to test that two populations with our measured *variances* have the same mean? How can we bootstrap or simulate this situation? These questions are difficult to answer, so that our recommendation is to use statistics and methods where you feel that you can answer them better.

The remaining methods are preferable in this way because they don't assume that we're looking for differences in any specific moment or feature of our data. Instead, they look for discrepancies between distributions in a more agnostic sense.

2.3 The χ^2 Statistic

Perhaps the most straightforward full-distribution comparison method is the χ^2-test. This test is only applicable to discrete (or binned!) distributions, but because of this, it will serve as a useful next step. Specifically, the χ^2-test (read as "**kai**-squred", not **chee**) is a method for testing the hypothesis that two frequency distributions are the same.

The distribution has a lot of theory behind it, but the general premise is to assume that you have some frequency distribution, say the expected number of each face of a fair die after 60 rolls, as shown in Figure 6. Because the roll of a die is a random event, we don't actually expect to get 10 perfectly of each event, but some fluctuation, as shown in the center column of Figure 6. So we consider the expected number of each die face to be E_i and we can look at the deviations of the observations, O_i:

$$Dev_i = O_i - E_i.$$

Let's say then that we don't really care about whether the observations were bigger or smaller than expectation, so we consider the squared deviations

$$Dev_i^2 = (O_i - E_i)^2.$$

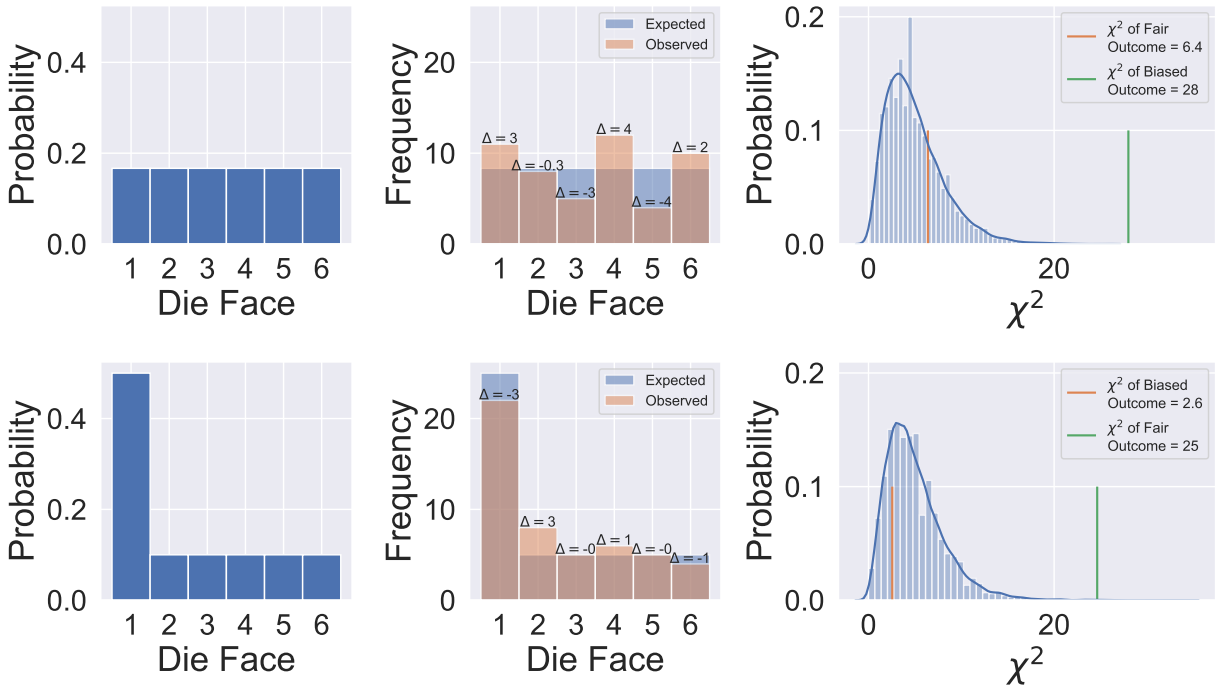


Figure 6: Two different dice, top row shows a fair die and the bottom a die that is biased towards 1. The left column shows the probability of any outcome, the center column shows the expected number after $N = 100$ rolls, while the orange shows the result of actually rolling 100 dice. The right column shows the χ^2 distribution generated by rolling dice with the same probability of 1-6 as the expected distribution, as well as the χ^2 values corresponding to the observations in the center column. Notice that the observations within the center column are consistent with the expectations, but the observations from the bottom row would be inconsistent with a fair die, and the top row would be inconsistent with our biased die.

We can then create a statistic θ that depends on O and E by summing up these squared deviations from expectation:

$$\theta(O, E) = \sum_{i=1}^{N} (O_i - E_i)^2. \tag{8}$$

We could then simulate a bunch of rolls from our fair die to generate a null distribution for this statistic, and we could then use this statistic to compare other die-distributions to see if their deviations are bigger or smaller than expected.

This statistic that we've created is *almost* the χ^2 statistic, but the χ^2 does better than ours by *scaling* the deviations by the expectation. That is,

$$\chi^2 = \sum_{i=1}^{N} \frac{(O_i - E_i)^2}{E_i}. \tag{9}$$

You can see why this is necessary by considering an expected distribution where the outcomes are not all equal, as shown in the bottom row of Figure 6. In this case, we can see that the odds of getting a 1 are way higher than 2-6, so a deviation of 2 should mean *less* for that outcome compared to the rest. Dividing by E_i scales each of the deviations so that they are comparable to one another.

We can then simulate / bootstrap / resample our data to generate a null distribution for deviations from our expected distribution. This is shown in the right column of Figure 6. Alternately, the χ^2 distribution has a known form:

$$f(x; \nu) = \frac{x^{\nu/2-1} e^{-x/2}}{2^{\nu/2}\Gamma(\nu/2)}, \tag{10}$$

where ν is the **degrees of freedom**. In the case of a die roll with 6 outcomes, we have $\nu = 6 - 1 = 5$ degrees of freedom. This is because for a given number of die rolls, say N, we can change any 5 of the frequency bins independently of the others, but the 6th bin will always be fixed by $N - x_1 - x_2 - x_3 - x_4 - x_5$. In general, when we have K possible outcomes for a random variable, the degrees of freedom involved in N simulations of that random variable will be $K - 1$. We can then use the theoretical CDF to calculate p-values as well.

As a final note, the χ^2-statistic, while intuitive and easy to calculate, is only appropriate for **discrete** distributions. You *can* apply the statistic to binned histograms, but as we'll see below there are more appropriate methods for continuous distributions. Additionally, the theoretical form of the distribution is only valid for frequency bins with more than 5 counts, so it's important to use both bootstrapping and theory to calculate p-values in those

cases.

2.4 $P(x < y)$

As a next approach to hypothesis testing, we can see from our drug trial example that often we are interested in whether a quantity is "less than" another quantity, whether those are two experimental conditions, or the difference between a null and an alternateive distribution. One way to do this is to consider the joint distribution of those two quantities that is formed by assuming the two distributions are *independent* and then summing up the amount of the joint distribution that is below the line $x = y$.

That is, if our two random variables are X and Y, then the part of the joint distribution below that line corresponds to the probability that X is less than Y, $P(X \leq Y)$. We can then say that if this probability is greater than some value that the likelihood that these two distributions are the same is small.

It's worth noting that if we have theoretical distributions, we can make this calculation computationally by breaking the domains of each distribution into a grid and adding up $P(X, Y)$ for all the grid points below the line $y = x$. On the other hand, if we have two empirical distributions, we can perform the following manipulation to make an exact calculation.

$$P(X < Y) = \int_{-\infty}^{\infty} \int_{y}^{\infty} P_X(x) P_Y(y) dx dy \tag{11}$$

$$= \int_{-\infty}^{\infty} P_Y(y) dy \int_{y}^{\infty} P_X(x) dx \tag{12}$$

$$= \int_{-\infty}^{\infty} P_Y(y) \left[1 - \bar{P}_X(y)\right], \tag{13}$$

where $\bar{P}_X(y)$ is the CDF of P_X evaluated at y. If we then discretize this into our x_i and y_j, we can write

$$P(X < Y) = \sum_{j=1}^{N_Y} P_Y(y_j) \left[1 - \bar{P}_X(y_j)\right] \tag{14}$$

$$= \sum_{j=1}^{N_Y} \frac{1}{N_Y} \left[1 - \bar{P}_X(y_j)\right], \tag{15}$$

because in a discrete distribution, the likelihood of seeing any one of your data points is $1/N_Y$. $\bar{P}_X(y_j)$ is then of course an empirical CDF evaluated at y_j.

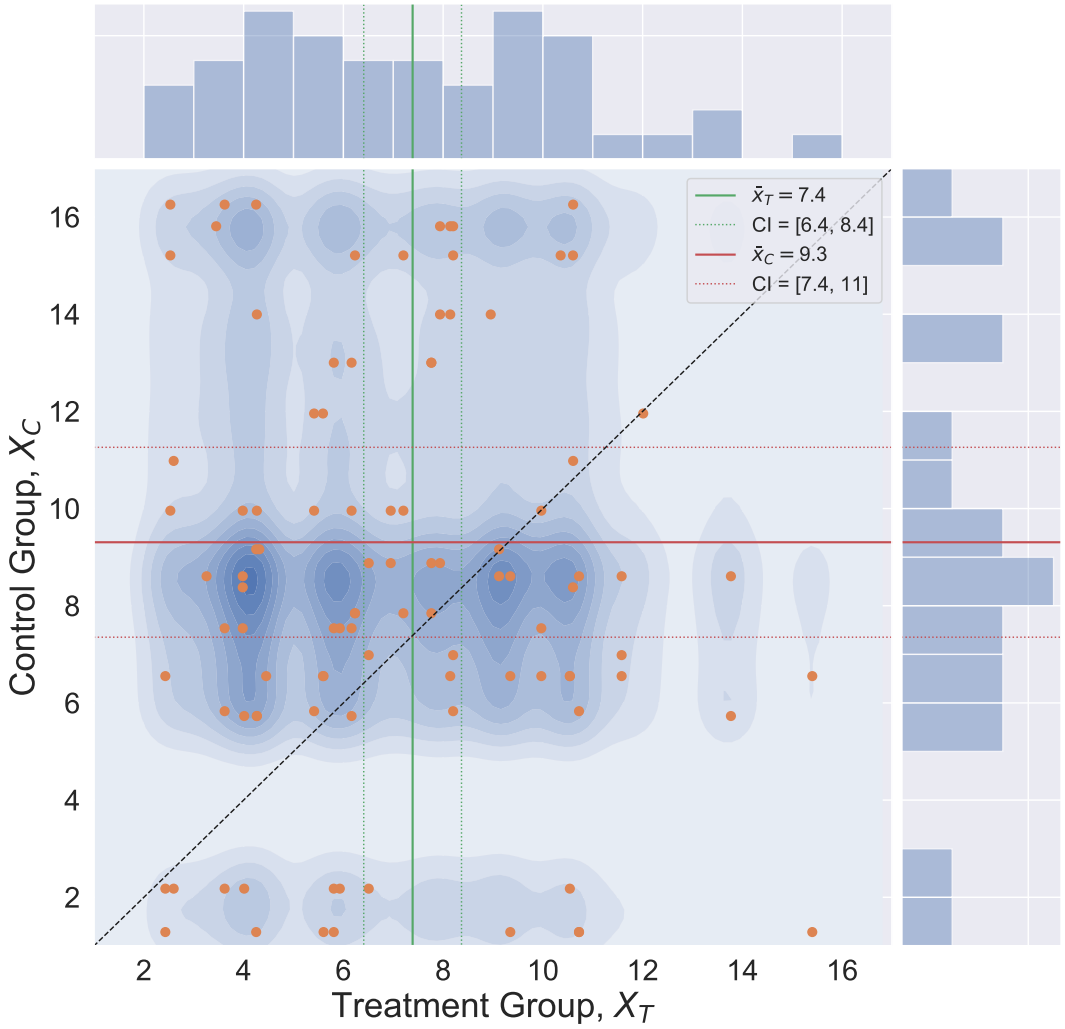


Figure 7: Estimate of the joint distribution of the X_T and X_C (treatment and control group) inflammation levels, assuming that they are independent of one another. The line $X_C = X_T$ is plotted in black, and the means and confidence intervals from Figure 1 are shown in green and red.

Of course, the simplest thing to do is to draw new samples from the marginal distributions, either by resampling or bootstrapping, and asking for the fraction of times that $X < Y$. In the drug trial data, as shown in Figure 7, comparing bootstrapped samples directly gives that $P(X_T < X_C) \approx 0.636$, while using Equation 15 gives 0.640.

Try It Yourself

You can use this information to try Problem 5 on Worksheet 3.2.

2.5 ROC Curves

Receiver Operating Characteristic curves (ROC curves) are diagnostic tools used to assess the performance of a **binary classifier**, i.e. a tool for putting data into two (binary) classes. This is relevant for our distributional comparison toolkit because we can frame the problem of distinguishing distributions as being a classification problem: given a value of a random variable, which distribution was it generated from, the null or the alternative hypothesis?

Consider Figure 8, where we can see in the left column that we might want to draw a line between the data and null distributions that maximally separates those distributions. More precisely, we want to find a line such that when we see a value to the left of that line, we assume the data came from the left distribution, and if it came from the right, we assume it came from the right distribution. We then want to pick the dividing line so that we are wrong the least often.

We can determine our ability to do this by varying where we put our line and comparing the CDFs at those line positions. This is shown in the right two columns of Figure 8, where the x-coordinate of the right column is given by the blue curve in the center panels, while the y-coordinate is given by the orange curves.

We can then calculate the area under these ROC curves, traditionally called **AUC** (Area Under Curve) to evaluate how well the classifier *could do* at the best threshold. If we have a perfect classifier, then there will be no overlap and we'll go from one PDF to another, drawing a square in the ROC space. On the other hand, flipping a coin will generate the black dashed line shown in the right panels of Figure 8. The AUC then quantifies this, with an AUC ≈ 1 being a very good classifier, while and $AUC \sim 0.5$ is basically just randomly picking. Not shown in the Figure, but entirely within your reach, you could then take one of your distributions, generate many bootstrapped sample and ask how often you see the AUC

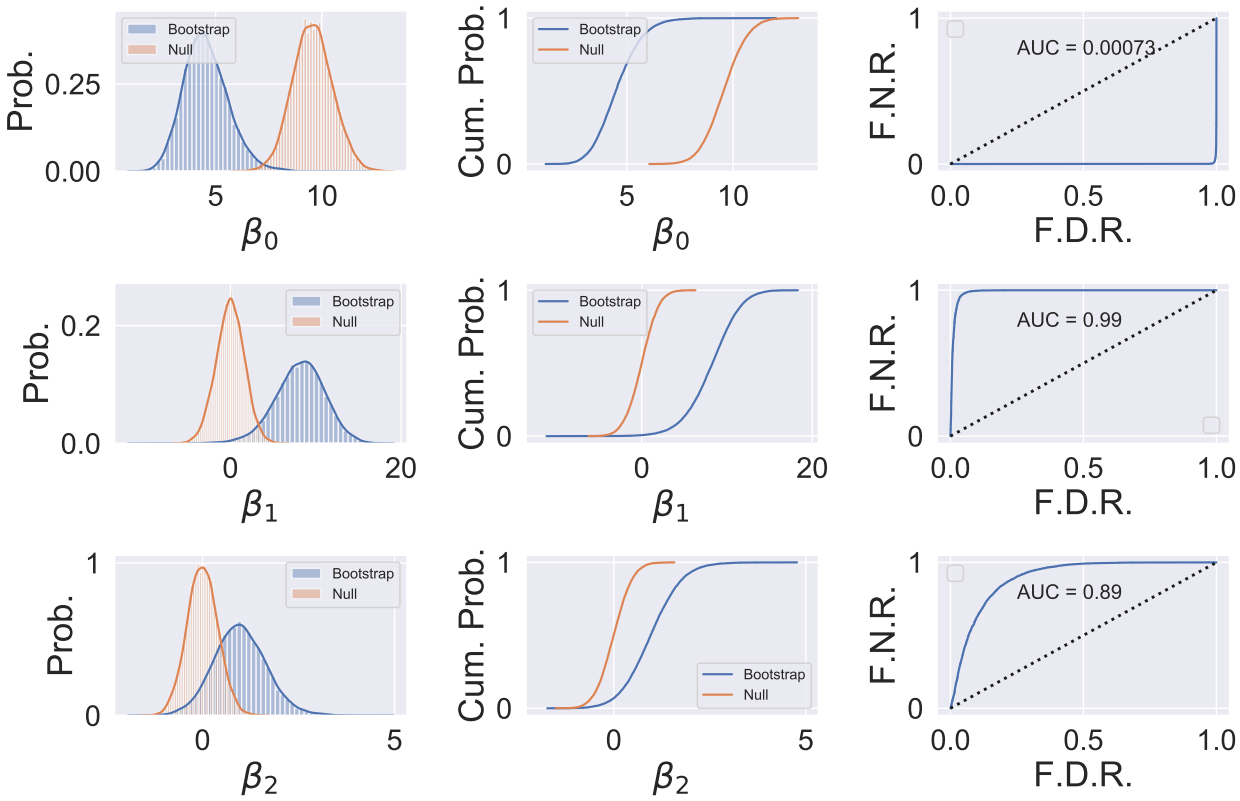


Figure 8: Generation of ROC curves from PDFs for each of the three regression coefficients involved in fitting the model $Rings = \beta_0 + \beta_1 x_{Height} + \beta_2 x_{weight}$. The left column shows the bootstrapped distribution with KDE estimates from `sns.distplot`. The center column shows the empirical CDFs, and the right column shows the ROC curve formed by plotting the data pairs $(P_{Boot}(B \leq \beta_i), P_{Null}(B \leq \beta_i))$. The area under the curve, or AUC, is then calculated as well. The "FNR" and "FDR" in the right column refer to the **False Negative Rate** and the **False Discovery Rate**.

from your data to estimate the likelihood of observing such a classifier by random chance.

Try It Yourself

You can use this information to try Problem 6 on Worksheet 3.2.

2.6 Non-Parametric Tests

While the previous two methods have been focused on whether the center most of the "mass" of a probability distribution is shifted in one direction relative to the comparison distribution (albeit in a more general sense than what the t-test was trying to assess), we noted earlier that distributional differences can sometimes be symmetric or more subtle. The following methods are based on quantifying the expected vertical fluctuations in an empirical CDF generated by a distribution. Because these methods are CDF-based, they are not biased towards detecting differences in one moment vs. another, but more generically detect unexpected divergences in any part of the distribution.

2.6.1 The Kolmogorov-Smirnov Test

The **Kolmogorov-Smirnov Statistic** or KS-statistic is the largest positive deviation between one CDF and another. When two distributions are different, this distance will be larger, even if this discrepancy shows up in a subtle way. This can be seen in the top panel of Figure 9.

You can probably figure out a way to calculate this yourself using eCDFs, but in Python you can calculate the KS statistic and it's p-value (2-tailed, watch out!) using `scipy.stats.ks_2samp`. When applied to the data in Figure 1, one obtains $KS = 0.266$, which is the same as shown in Figure 9, but the p-value is assessed at 23.3%. This is likely due to the fact that the p-value in the figure does not correspond to a 2-tailed test; the distributions were not reversed in the bootstrapping process.

This statistic and test is extremely powerful, and is an excellent first place to start when comparing distributions.

Try It Yourself

You can use this information to try Problem 4 on Worksheet 3.2.

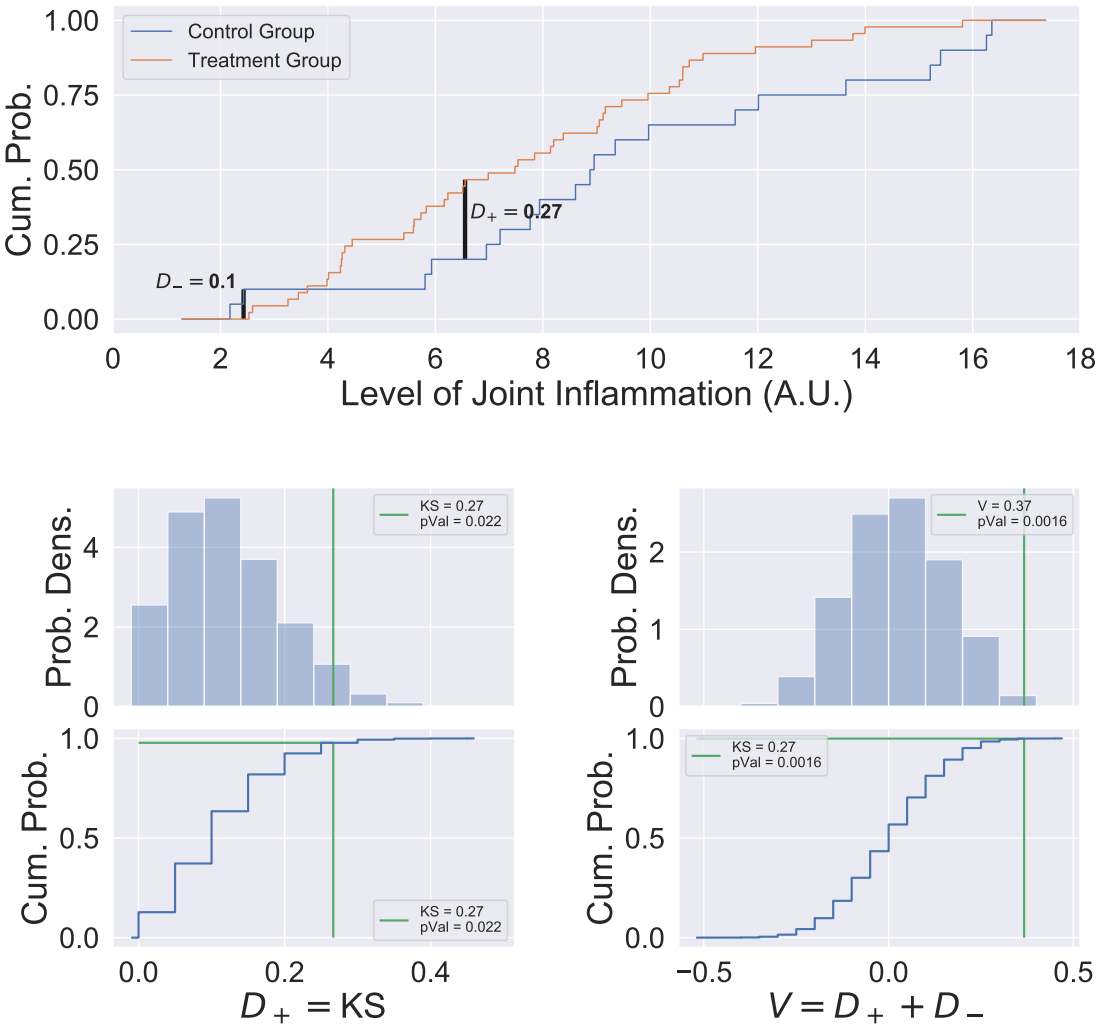


Figure 9: Illustration of the KS and Kuiper Statistics. The top panel shows the D_+ and D_- largest positive and negative distances between the two eCDFs. Bootstrapping the control group and measuring D_+ and $V = D_+ + D_-$, we get the distributions and eCDFs shown below. The empirical p-values for the actual data are shown on the plots as green lines.

2.6.2 The Anderson-Darling Test

However, some statisticians noted that the KS-stat necessarily ignored deviations in CDFs that are near the tail. Since all CDFs have to start and end at 0 and 1, respectively, there isn't as much room for them to diverge from other distributions in those areas. This is where the **Anderson-Darling**-statistic comes in. This statistic has a slightly more complicated formula, which is not worth belaboring, which is designed to bias the assessment of divergence equally towards the tails and the center of the distribution. As a result, it is generally regarded as more robust than the KS-statistic.

In Python, one can calculate the AD statistic using `scipy.stats.anderson_ksamp` function. For the drug trial data, we got that the statistic had a value of 1.817, which corresponded to a likelihood of ∼5% that the two distributions were the same.

2.6.3 The Kuiper Test

Finally, another perturbation to the KS test that is easier to calculate than the AD statistic is the Kuiper statistic. Named after the same Kuiper who names the Kuiper Belt, this statistic naturally overcomes some of the deficiencies of the KS test, particularly with regards to **circular data** (data that exist on a periodic domain, like time or cell cycle phase). Calculations of this statistic from bootstrapped samples are shown in Figure 9, where we see that even a 2-tailed p-value suggests that the data are unlikely to occur by chance.

2.7 Information Theory

2.7.1 Maximally uninformative distributions

This section might seem like a bit of an aside. We ask for your patience. Consider the scenario where all you know about your null, your control population, or any hypothesis that you wish to confront your data to is just a few of its properties. And, as has been the theme of this entire module, confront means you wish to assess what the likelihood of your data is were a particular hypothesis true. A trivial example might be that you know, say, the mean of an quantity under a null hypothesis, and nothing else about the distribution. How do you now confront such a minimal null hypothesis to distribution data? The maximum entropy approach suggests the following: Construct the distribution with most uncertainty that agrees with the few features you have prior knowledge of. This distribution is referred to as the maximum entropy distribution and is a route to constructing a null distribution

when you have sparse information. Note that the idea of maximum entropy is used in a large class of situations, not just null hypothesis generation. One aother relevant context is in the case of Bayesian data analysis. Perhaps you recall the arbitrary nature of coming up with a prior in the Bayesian parameter estimation section. While we demonstrated that in the limit of large data the initial differences in distinct priors is inconsequential, it was also clear that with limited data your choice of prior matters. How does one deal with this seemingly subjective nature of taking a Bayesian point of view?

In order to try and resolve thess conundrum we must introduce the idea of entropy and a little on information theory.

2.7.2 Information Theory: A primer

The concept of information is too broad to be captured completely by a single definition. However, for any probability distribution, we define a quantity called the entropy, which has many properties that agree with the intuitive notion of what a measure of information should be. This notion is extended to define mutual information, which is a measure of the amount of information one random variable contains about another. Entropy then becomes the self-information of a random variable. Mutual information is a special case of a more general quantity called relative entropy, which is a measure of the distance between two probability distributions. All these quantities are closely related and share a number of simple properties. For those who know a little bit about Thermodynamics and Statistical Mechanics will recognize both the word, and functional form, of Entropy as defined in Information Theory.

Perhaps the most important concept/definition in information theory is that of entropy: a measure of uncertainty of a random variable. We start with a probability distribution, or mass, function $p(x) = Pr(X = x)$. X is a random variable, and x is a specific outcome. So for example, if we are tossing a coin then $p(heads) = Pr(Outcome of single coin toss = heads) = 0.5$, if the coin is fair. The outcome of a single coin toss is the random variable, "heads" is a particular outcome. The above definition kind of only makes sense for a discrete random variable. What do I mean? If I asked you what the probability of finding someone in this building with mass = 76.34674 kg, it would be equal to 0. In fact the probability of finding a person of any particular weight is 0! What makes sense is to then ask what is the probability of finding someone with mass between 70kg and 71kg. Mass is therefore a continous variable and requires a slightly different viewpoint. That said, any measurement made has an accuracy associated with it. Even if we weigh someone, our scales (even the best ones! – eventually you are constrained by the accuracy set by quantum mechanical

$$X = \begin{cases} 1 & \text{with probability } p, \\ 0 & \text{with probability } 1-p. \end{cases}$$

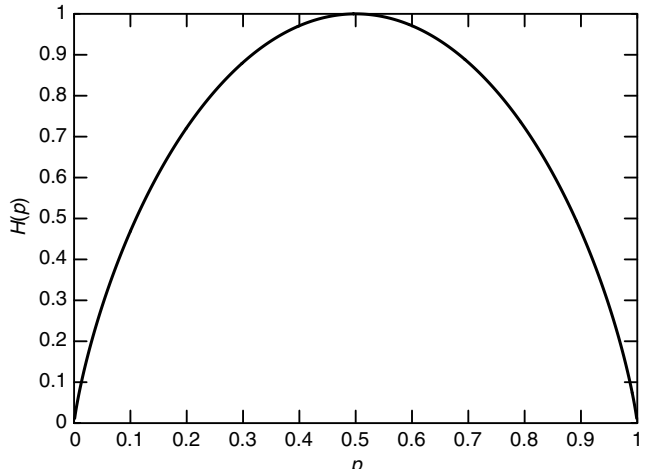


Figure 10: The entropy of a single coin toss as a function of the parameter p, the probability of heads

uncertainty) have some resolution, so when we say we measure someones weight and it is 75.8kg, we really mean 75.8kg ±0.05kg, for example. And that of course has a probability since its a non-zero sized interval.

Lets stick with discrete random variables for now, with the knowledge that once you take accuracy of any measurement into consideration even a continuous measurement is really a discrete one. The entropy $H(X)$ of a discrete random variable X is defined to be

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) \log p(x)$$

The log is in base 2, and entropy is expressed in bits. For example, the entropy of a fair coin toss is 1 bit. Mathematically speaking, we calculate the entropy by summing $plogp$ over all possible outcomes of the experiment, in this case there are just two outcomes, heads and tails: $H(X) = -(1/2log(1/2) + 1/2log(1/2)) = -(-1/2log(2) - 1/2log(2)) = -(-1/2 - 1/2) = 1$. Intuitively, a bit is a 0 or a 1, and so it quantifies the outcome of an experiment (a realization of a random variable) with two choices (in this case, heads and tails). What is the entropy of the random variable associated with the roll of a dice? $H(X) = log(6) \approx 2.58$ – do the calculation yourself! It is worth stressing that the entropy is a function of the distribution of X. It does not depend on the actual values taken by the random variable, only on the probabilities.

Why this definition? First, lets just study its properties taking for granted that I am not teaching you something entirely uninteresting.

Consider Figure 10. What we are doing is simply plotting the entropy of a single

$$X = \begin{cases} a & \text{with probability} \frac{1}{2}, \\ b & \text{with probability} \frac{1}{4}, \\ c & \text{with probability} \frac{1}{8}, \\ d & \text{with probability} \frac{1}{8}. \end{cases} \quad H(X) = -\frac{1}{2}\log\frac{1}{2} - \frac{1}{4}\log\frac{1}{4} - \frac{1}{8}\log\frac{1}{8} - \frac{1}{8}\log\frac{1}{8} = \frac{7}{4} \text{ bits}$$

Figure 11: An arbitrary discrete random process

coin toss as we vary the probability of getting a head, p. As intuition dictates, when p=0 or 1 there is no uncertainty in the outcome and hence no information. And clearly, the uncertainty peaks when p=1/2 since the outcome of the experiment is maximally uncertain.

Another example can be seen in Figure **??**. Suppose we have a random variable that can take on 4 values with different probabilities as described below. Suppose that we wish to determine the value of X with the minimum number of binary questions. An efficient first question is Is X = a This splits the probability in half. If the answer to the first question is no, the second question can be Is X = b The third question can be Is X = c The resulting expected number of binary questions required is 1.75. This turns out to be the minimum expected number of binary questions required to determine the value of X. So that is how you should think about information, its the minimum expected number of yes/no questions you have to ask to figure out the value of a random variable.

An important definition is the joint entropy H (X, Y) of a pair of discrete variables (X, Y) with a joint distribution p(x, y) is defined as

$$H(X, Y) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x, y)\log p(x, y),$$
$$H(X, Y) = -E\log p(X, Y).$$

We now introduce mutual information, which is a measure of the amount of information that one random variable contains about another random variable. It is the reduction in the uncertainty of one random variable due to the knowledge of the other.

Consider two random variables X and Y with a joint probability mass function p(x, y) and marginal probability mass functions p(x) and p(y). The mutual information I (X; Y) is the relative entropy between the joint distribution and the product distribution p(x)p(y)

$$
\begin{aligned}
I(X;Y) &= \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log\frac{p(x,y)}{p(x)p(y)} \\
&= D(p(x,y)||p(x)p(y)) \\
&= E_{p(x,y)}\log\frac{p(X,Y)}{p(X)p(Y)}.
\end{aligned}
$$

So whats the relationship between entropy and mutual information?

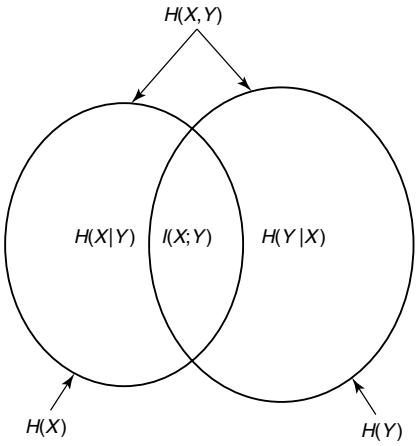
The mutual information I (X; Y) is the reduction in the uncertainty of X due to the knowledge of Y.

By symmetry, it also follows that the mutual information I (X; Y) is the reduction in the uncertainty of Y due to the knowledge of X.

And so a symmetric way of writing mutual information in terms of entropy is

$$I(X;Y) = H(X) + H(Y) - H(X,Y).$$

The following Venn diagram helps when thinking about the above definitions



While the definition of mutual information is straightforward there are all sorts of ways of estimating its value in the context of finite amounts of data. Which technique one should use depends on the specific context of the calculation.

2.7.3 Maximum Entropy Distributions

Now that you have some sense of what entropy is we can introduce the principle of maximum entropy. According to the principle of maximum entropy, if nothing is known about a

distribution except that it has some specified properties or features, then the distribution with the largest entropy consistent with those properties should be chosen as the least-informative default. The motivation is twofold: first, maximizing entropy minimizes the amount of prior information built into the distribution; second, many physical systems tend to move towards maximal entropy configurations over time. This second point, regarding physical systems is something you have encountered in disguise – the principle that systems flow until it minimize its free energy.

Said another way, entropy underlies a core theory for selecting probability distributions. Thomas Jaynes argues that the maxent distribution is "uniquely determined as the one which is maximally noncommittal with regard to missing information, in that it agrees with what is known, but expresses maximum uncertainty with respect to all other matters". Therefore, this is the most principled choice.

Seen from a slightly more mathematical lens, the features (the aspects of the pdf that are known), can be seen as constraints on an optimization scheme that attempt to maximize entropy.

What do some solutions of this principles look like?

1. What is the maximum entropy distribution over a finite (discrete) range $\{0, 1, ..., N\}$? $p(x) = \frac{1}{N+1}$.) This is the uniform distribution. In Bayesian estimation, this means that we select a uniform prior when we know nothing about the source of our data (other than the range of possible outcomes), yielding a maximum likelihood estimation. Recall the MLE is unbiased; maxent echoes this property.

2. What is the maximum entropy distribution for a positive continuous random variable X with mean μ? $p(x) = \frac{1}{\mu}e^{-x/\mu}$. This is the exponential distribution you have already encountered in your analysis of bacterial chemotaxis data. Its also the distribution you get for radiactive decay, and a host of other natural phenomena. What it means is that if you nothing other than the mean value of a random variable then the most noncommital, maximally uninformative, prior or distribution for the process is an exponential one.

3. What is the maximum entropy distribution with mean μ and variance σ? $p(x) = \frac{1}{\sqrt{2\pi\sigma^2}e^{-(x-\mu)^2/2\sigma^2}}$. This is the Gaussian/Normal distribution. Again, if all you know that a random process has a particular mean and variance then the Gaussian distribution is most noncommittal distribution that encodes just that information and nothing else.

These are truly beautiful results.

Of course, the theme of this course is not to remember names and formulas for distributions. But focusing on big practical concepts is the goal of this course. Using the

principle of maximum entropy as a guide to determining Bayesian priors is incredibly useful. Furthermore, given that you can numerically solve for the maximum entropy distribution consistent with any set of constraints, generating null distributions using this approach can also be powerful.

3 Chapter Review

What was this module about? In short, I believe this is where the rubber really hit the road. You saw how for the first time all the concepts and methods introduced in module 1 and 2 can really be used to conclude hypotheses from data.

The most important concept, indeed the one that was repeated over and over again, is 1) to identify a null hypothesis that is a negation of the effect you are looking for, 2) define a statistic, a number that parametrizes/quantifies distances in a manner that you deem appropriate for your data, 3) Generate a distribution for the statistic consistent with your null, 4) obtain a p-value for your data by looking up empirical null distribution of the statistic, 5) Generate a distribution of p-values, as outlined above, through bootstrapping your data. 6) Repeat the procedure using multiple statistics to prove the robustness of your conclusions.

This protocol identifies that there are two crucial steps in hypothesis testing that require you as the data analyst to make choices: 1) What is the null hypothesis that is the quantitative negation of the effect of interest, and 2) What statistic will you use? I hope we have made clear through introducing a large class of statistics that there is no correct choice. Each statistics has is strengths and weaknesses. Ideally, the data analyst uses as many as possible to demonstrate the robustness of their conclusions. The issue of defines a quantitative null is truly an art. There are some guidelines, but really this expertise is built through experience. Too naive a null and you will get tiny p-values not because the effect was strong but because you chose a poor null. Regardless, the one constant theme in this module is that you need not restrict yourself to statistics and nulls that admit a closed form solution that the classical statistician limited themselves to. You have a computer and so can use the incredibly powerful approach of bootstrapping to free yourself of using a restricted set of tests and null distributions – you can make your own that suit your specific needs.

More broadly speaking, we believe that this is the first module in which you are gaining insights into thinking statistically about real world problems where you not only need to be quantitative but need to make decisions based on your data. At the heart of being able to make that decision is to think statistically. To think statistically, in our view, is to adopt a distributional view of data, where you need to not only produce point estimates of features of the distributions and statistical quantities of interest, but also intervals of confidence.

4 Details and Code

Chapter Five: Model Selection and Dimensionality Reduction

Eric Johnson and Madhav Mani

May 11, 2020

Throughout these notes we have gone from thinking about data probabilistically (Module 1), to fitting models (Module 2), to testing models against null hypotheses (Module 3). At each step of the way there was a model or equation attached to each set of data, and we continually put off the question of *how to obtain this model*. In this chapter we will make one last dent in this problem by introducing some methods for **model selection**.

We want to note that model *selection* is a subset of the bigger problem that we might call model *generation*. However, model generation is in some ways *the problem* that science is trying to answer: if we make some observations, how are they related and how can we predict future observations? In this way, if we had a foolproof methodology for coming up with the best model when confronted with any data, we'd be very famous and science would be easy. Obviously this is not the case, and so we will refrain from discussing the ins and outs of how to come up with a novel model or class of models, but instead restrict ourselves to the problem of how to choose between a set of predetermined models. This is the problem that we call model selection.

This isn't to say that we're going to restrict ourselves to choosing between two or three obviously different models. Our classes of models might be the set of all polynomials or all linear combinations of a set of observations. In particular, as the sizes of your data and problems become larger, the class of even simple (linear) models can become quite large. To handle these cases, we will also use this chapter to introduce some methods in **dimensionality reduction**. These methods are centered on determining how to reduce the size of your problem in a meaningful way so as to allow for more justified and robust models.

This chapter will first elaborate on some general principles of model selection, then spend a moment discussing the Bayesian approach to the problem. We will then introduce **Principal Component Analysis** (PCA) and **LASSO** as some canonical (and useful!) di-

mensionality reduction methods. We will conclude by discussing a few other methods for model selection that are sometimes used, but that we feel are less robust than the dimensionality reduction methods.

1 What is Model Selection?

Ok, so you've got your set of models and you want to pick the "best" one to base your analysis on, how do you do this? Well first let's unpack what we might mean by "best". "Best" might mean that the model has the smallest average residuals compared to data (it fits the data best), it might mean that it has the highest predictive accuracy when presented with new data, it might mean that it is the most interpretable in terms of the physical properties of your system, or "best" could mean that you have the highest confidence that the model parameters are non-random. There are also many other yardsticks that we could come up with to define "best."

You might be able to imagine that for a single data set, different models might be described as "best" depending on what criteria we use. For example, a high-order polynomial might fit every data point exactly and have no residuals, but a linear model might be more predictive. A thermodynamic model might be physically interpretable, but a simpler function might be more robustly non-random. As with everything we've talked about in these notes, the methods that we pursue will depend on these subjective choices that we make about our goals, so we should make sure that we are explicit with our choices so that our conclusions can be assessed properly.

Going forward, we will alternate between using predictability and minimal residuals (best fitting) as our main quantitative criteria. Predictability is a useful metric for "best" because it averages over the randomness in how we collected our data and ensures that the model is useful. Asking for minimal residuals is also a good heuristic because, as we've noted, this corresponds to maximizing a likelihood or posterior function, which gives us probabilistic confidence in our models. In many cases we will attempt to optimize both these quantities, but we will generally focus on just one at a time.

In this way, we define model selection as the process of choosing i such that the model $\mathcal{M}_i$ minimizes or maximizes some quantity of interest over the set of all models, $\{\mathcal{M}_i\}$. It's worth noting that if we consider our set of models to be the set of models with the same functional form but different parameters, then model selection is exactly the same as parameter estimation! That is, if our set of models is $\{y = \beta_i x\}$, where each model has a different β_i, then picking the best model amounts to estimating the best β_i, which you

already know how to do!

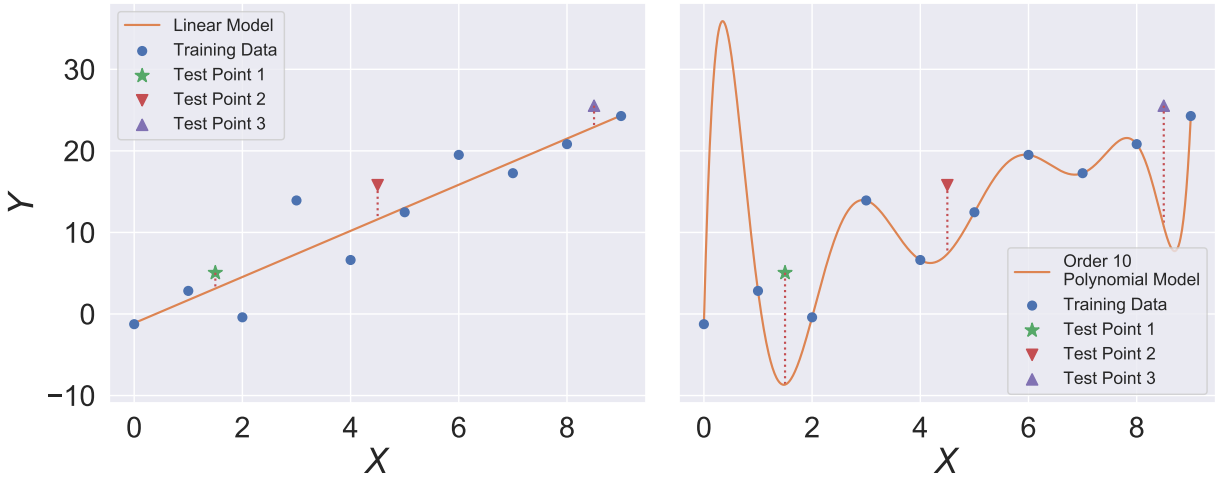


Figure 1: Example of two polynomial fits to the same data set. The left panel shows a linear fit to the data, while the right shows a 10^{th} order polynomial fit. The mean squared residual for the linear model is 9.76 while for the polynomial fit it is **0**. But the average squared prediction errors are 9.38 and **159.50** for the two models, respectively.

So what do we do if our models have different forms? What if we want to choose between a statistical model and some graph-based dynamical model? The simplest thing to do is to compute the mean of the residuals when the models are fit, since a better fitting model is better, right? As an example, consider the two models in Figure 1. In this case, the linear model is a worse fit, while the 10^{th} order polynomial fits the data exactly, so at first pass we might say that the 10^{th} order polynomial is the better model.

But you should be skeptical of this assessment, the data "look" linear, and there's no evidence of the wild fluctuations of the polynomial model, so how can we assess whether these models are really any good? As mentioned above, another thing we can do is ask how good they are at making *predictions*. We can do this in a couple ways, but the most comprehensive and straightforward is to do Leave-One-Out Cross Validation (LOOCV). As a reminder, this means that we leave out one data point, fit the model to the rest, and ask how well we predict the left out data. When we apply this technique to the linear and polynomial models in Figure 1, we find that the median squared prediction error for the linear model is 2.59 and for the 10^{th} order polynomial is **1089.14**! Thus we can say that while this polynomial fits the data perfectly, it does not do so very robustly and as a model it seems to be prone to **overfitting**.

We'll explore this more later on, but in general there is a tradeoff between model

simplicity and model robustness. This is commonly known as the **bias-variance tradeoff**, and we'll look into it more closely as an aspect of dimensionality reduction.

This example should hopefully convince you that looking at either of our simple metrics alone will not provide an assessment of model "quality," but instead we will have to try to balance several objectives at once. If we want to consider other criteria for selecting models, we will similarly have to balance them against each other. The remaining part of this chapter will be devoted to elaborating on methods that allow us to strike this balance in a rigorous and quantitative fashion.

2 Bayesian Model Selection

As we have made clear throughout these notes, there is a Bayesian perspective on most everything, and it is usually a useful perspective, even if it is not always the most practical methodology. In this case, we will again (ab)use the flexibility of Bayes' Theorem to explicitly write down the quantity that we want and then we will unpack how we can get there computationally or theoretically.

In this case, let's say we have some data and we want to choose a model that best describes this data. Then if we want to assess the probability that a given model is responsible for observed data, we can use Bayes' Theorem to write down

$$P\left(\mathcal{M}|\, X\right) = \frac{P\left(X\,|\mathcal{M}\right) P\left(\mathcal{M}\right)}{\sum_{M\in\mathbb{M}} P\left(X\,|M\right) P\left(M\right)}. \tag{1}$$

Here, $\mathcal{M}$ is our specific model of interest, X is our data, and $\mathbb{M}$ is the set of *all models*.

You should note that Equation 1 is *correct*, but not necessarily *useful*. For example, how can we assign a prior to a *model*, and how can we perform a sum over *all models*, $\mathbb{M}$? The short answer is that we will not, in general, attempt to compute the denominator of Equation 1. Instead we will compute Equation 1 up to a constant of proportionality or compare *posterior ratios* of two models. Additionally, it is common to simply set the prior distribution to be uniform over all models. That is, the naïve statement is that all models are equally likely, which you should understand is not correct, but is the best guess according to information theory.

In this way, if you accept that we've addressed some concerns about Equation 1 by ignoring the denominator and assuming a uniform prior, then we can simplify the problem

to the following:

$$P\left(\mathcal{M}|\, X\right) \propto P\left(X\,|\mathcal{M}\right). \tag{2}$$

We can use properties of probabilities and the concept of *marginalization* to write

$$P\left(X\,|\mathcal{M}\right) = \int P\left(X\,|\,\Theta,\,\mathcal{M}\right) P\left(\Theta\,|\mathcal{M}\right) d\Theta, \tag{3}$$

where Θ are our model $\mathcal{M}$'s parameters. In this context, the quantity in Equation 3 is called the **model evidence**. The concept of model evidence as calculated with Equation 3 is that for a given model, there may be many parameter combinations that make sense, so we should consider the data's likelihood of arising from our data as being the average likelihood given some specific parameters *weighted by how likely those parameters are.*

As an example[1], consider a coin that has been tossed 200 times and that turned up heads 115 times. From earlier chapters, we know that the likelihood of this occurring is given by the binomial formula, $P_B(115|200, p) = \binom{200}{115} p^{115}(1-p)^{200-115}$. The question then is, which is a better model: assuming that all coins are exactly fair, $p = 0.5$, or assuming p has some specified prior distribution.

Let's call $\mathcal{M}_1$ the model where $p = 0.5$. Then we have that

$$P(X|\mathcal{M}_1) = P_B(115|200, 0.5) = \binom{200}{115}(0.5)^{200} \approx 5.96 \times 10^{-3}.$$

On the other hand, let $\mathcal{M}_2$ be the model where p is distributed like a beta distribution with $a = b = 10$. (Verify for yourself that this is a distribution centered on $p = 0.5$ with a width of $\sim$0.3.) This might represent some thought that coins are on average fair, but there are some flaws introduced in the minting process. We then get that

$$\begin{aligned} P(X|\mathcal{M}_2) &= \int_0^1 P_B(115|200, p) \cdot \frac{p^{a-1}(1-p)^{b-1}}{\mathcal{B}(a,b)} dp \\ &= \binom{200}{115}\left(\frac{1}{\mathcal{B}(a,b)}\right)\int_0^1 p^{115+a-1}(1-p)^{85+b-1} dp \\ &= \binom{200}{115}\left(\frac{1}{\mathcal{B}(a,b)}\right)\frac{(115+a-1)!\,(85+b-1)!}{(200+a+b-1)!} \\ &= 1.39 \times 10^{-2}. \end{aligned}$$

(Don't worry about this math, just consider the conclusion and note that this problem is *solvable.*) This result suggests that there is nearly two and a half times more evidence for

[1] Thanks to Wikipedia for the great starting point!

the model with a bit of wiggle room rather than the perfectly fair coin model.

Try It Yourself

Look up the formula for the beta distribution, as well as how to use `scipy.special.beta` and `scipy.special.comb`. Set $a = b$ in the formula above and examine how the model evidence changes as a increases using the above formula. Use `scipy.stats.beta` to examine what the prior looks like for different values of a (and/or b).

We can note that if we have only two models, taking the ratios of the posteriors, and therefore the ratios of the model evidences gets rid of the denominator from Equation 1. That is, instead of ignoring the denominator as a constant factor, this also works to avoid explicitly calculating the denominator that works even if we don't have a finite set of models. This ratio is called the **Bayes Factor** and is a common metric for evaluating the relative likelihoods of different models. In the present case, the Bayes factor of 2.34 would be *substantial*, but not overwhelming, evidence for the "usually-fair" coin model.

In the present case, it seems that the fair coin model is less likely, which may be surprising. However, if we consider what's happening in the integral of Equation 3, in the model that allows for some variance in the fairness of a coin, we're weighting the model parameters *towards* the evidence (115 heads out of 200 tosses) fairly frequently, so there are many more "good" models in our average than the simple binomial model. We can see that if we relax our prior to the case $a = b = 1$, which is a uniform distribution for p, that the evidence becomes 4.98×10^{-3}, which is slightly less than our fair coin model. This preference for simpler models is a useful feature of Bayesian model selection and is often cited as a justification for the application of Occam's Razor. (In some ways, assuming that $p_{Heads} = 0.5$ is simpler than considering all options for p_H; similarly, assuming that $p_H \approx 0.5$ is simpler than prescribing it to be one value.)

In general, Bayesian model selection can be very useful for distinguishing between a few models in a very rigorous way. It has the added feature of naturally preferring simpler models, which can make it a good balancer in the bias-variance tradeoff.

Try It Yourself

Compute Bayes Factors to assess the likelihoods that the Frequentists were cheating in Worksheet 3.1. That is, now you can assess the relative likelihoods of the different priors proposed by the Frequentists and the Bayesians. Which model has more evidence?

3 Dimensionality Reduction

While we've spent a lot of time in these notes discussing the situation where you don't have many (N) samples, we haven't done a ton to address the very relevant case where you also have many (K) observations. In such a situation, model formation and even basic parameter inference can be made very difficult by the **curse of dimensionality**.

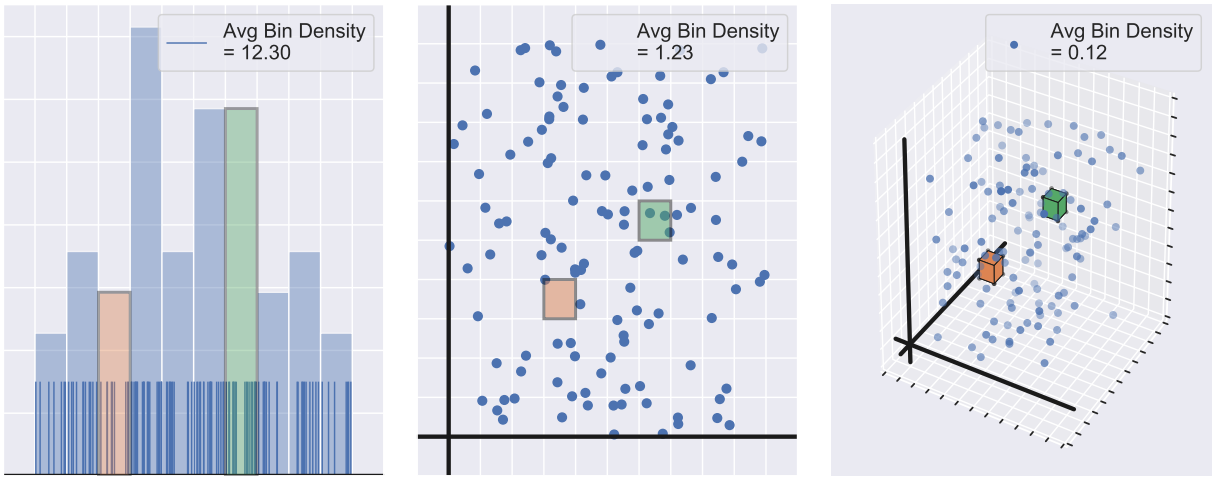


Figure 2: Illustration of the Curse of Dimensionality. Each panel displays the same number of data points. The highlighted cells illustrate the average number of points in any given region.

The curse of dimensionality refers to the notion that the density of data points decreases exponentially with the number of dimensions in which the N data points exist. That is, if I need an average of 10 points per unit length to make a good estimate in one dimension, as in the left panel of Figure 2, then I will need another $10L$ data points if I add another dimension of size L in order to make the same quality of estimate. Adding yet another dimension will require another $10L$ data points, and so on for each dimension I add. Figure 2 shows what happens when the number of data points stays constant but dimension increases: each "bin" (unit of space) holds a decreasing number of data points, so any estimates of quantities in that region of space are correspondingly weaker. If we need M data points per dimension, then the number of data points needed to preserve our estimation ability is given by M^K, where K is the number of dimensions. This obviously becomes astronomical as K increases, so that even in the best case scenarios we are often in the data-poor regime.

More philosophically, it's worth spending a moment to think about whether we really live in the era of "big data." On the one hand, researchers are increasingly making novel

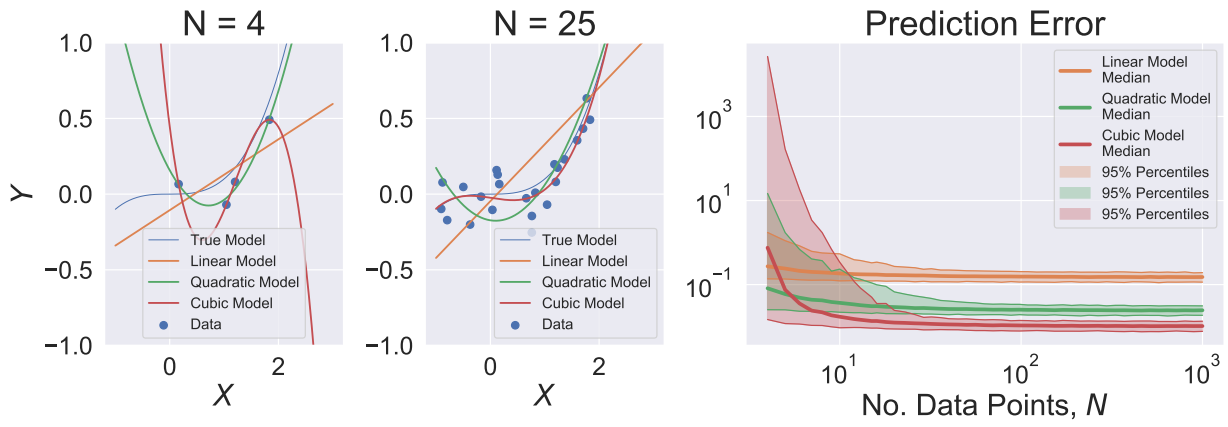


Figure 3: Illustration of how certain amounts of data are required to capture features in data. In the left panel, 4 data points are generated by adding noise to the model shown in blue. The orange, green, and red lines are linear, quadratic, and cubic fits to the data, respectively. The center panel is the same as on the left, but with $N = 25$ simulated data points. The right panel shows how the distribution of prediction errors change as the number of data points increases. The prediction error shown here is the mean squared residual on 100 test points. The median, 2.5^{th}, and 97.5^{th} percentiles of 1000 bootstraps of this calculation are shown.

measurements of myriad phenomena such as genetic sequences or social media dynamics. But the alternate view that we urge you to consider is that every new measurement adds a new axis to your graph, and the curse of dimensionality *guarantees* that you will have sparser sampling even if you have many observations. Since the point of collecting data isn't just to store lists of numbers, but to discern trends and actually *learn* something, you should be extremely concerned about how well your data are spread throughout space.

If you're thinking practically, you might want to know how dense your sampling needs to be to give reliable conclusions. You should then be comforted to know that there is the Nyquist-Shannon Sampling Theorem that tells us that if the smallest changes in the data happen on a scale $1/L$, then we need average sampling of $2L$ to recover the "signal." More intuitively, consider Figure 3 where data from a somewhat bendy model is fit by functions with varying capacity. The theorem suggests that the more that your underlying signal changes – in this case as a cubic function – the more data you need to make a robust fit.

In the example in Figure 3, the linear, quadratic, and cubic models all have roughly similar performance when there are just a few data points. (In fact, the cubic model is penalized for making predictions that aren't born out by the data!) The right panel of Figure 3 shows that the performance of the cubic model only starts to completely outperform

the quadratic model when $N > 25$ data points are added, where a specific example of this ambiguity is shown in the center panel. In this case we control all the parameters of the system, so it's easy to say that we should use the cubic model, but in practice, it may be hard to assess that you have enough data to detect the difference between a quadratic and cubic function.

However, it's the point of this course to illustrate that you now have some tools with which to make this call. The right panel of Figure 3 shows how you can use bootstrapping and training/test splits to assess whether your model's performance is a result of overfitting, as an example. You can hopefully intuit that as you move your problem to higher dimensions, you will be encountering this problem along *every* axis, so being aware of these issues is a requirement for a robust analysis.

In any case, as a result of this phenomenon of the curse of dimensionality, we often want to determine whether our data really occupy the entirety of our high-dimensional space or just some smaller subset. This is what dimensionality reduction refers to: the attempt to reduce the number of dimensions of the problem. If we can perform this successfully, then we can improve the quality of any analysis or estimates by working in the reduced space.

Is this justified? Well, even though we are making a large number of parallel measurements, we don't believe that the variation in the data lives in that high-dimensional space – the different measurements (axes in the high-dimensional space) are *correlated*. Said another way, the observations in your data set aren't independent of one another. This means three things:

1. The true dimensionality of your data is smaller than that of your measurement space
2. Estimates and conclusions drawn from lower-dimensional spaces will be better sampled and potentially more statistically robust
3. The "natural" parameterization of your data may not be your measurement space

Point 3 is especially relevant because it's also worth noting that we often want to reduce dimensions for modeling reasons, specifically, if we think that a system or phenomenon is governed by only a few variables, we should seek to capture those dimensions and not others. For example, if I could know the position and velocity of every molecule in a gas, it would still be a better model to use the Ideal Gas Law ($PV = nRT$) than all those positions

and velocities individually. In this way, dimensionality reduction can also help us develop *interpretable* models.

In the rest of this section, we will first introduce the formal definitions for correlation and covariance, then the methods of **PCA** and **LASSO**. We spend some time discussing correlation because it will help you develop some intuition for what this course is used for, but also because PCA is directly built on those formalities. Throughout this section, we will try to privilege intuition over mathematical formulas, but some precision is required in places. We will also end each introduction with guidance on how to *use* these methods in the computational and statistical way that we have advocated in this class (there will be a lot of bootstrapping!).

3.1 Correlation and Covariance

To this point, we have sometimes discussed *relationships* between data. Indeed the previous module was centered on testing the evidence behind a specific module: whether a specific relationship is visible in the data or is just random noise. Often in making these tests, we asked you to first *look* at your data and use your visual intuition to guide your guesses, but as we pressed on, that became less possible. Now we're discussing the very modern data that have dozens or hundreds of different observations, so that looking for trends by eye isn't possible. To this end, it is useful to try and be more precise about what it is we're looking for when we're doing this visual inspection. This is where the mathematical definition of **correlation** comes in.

To build intuition about the formulas we'll be giving, let's consider an example. Imagine going outside and it's sunny out. On an average day, is it more likely that it's warmer or that it's cooler, given that it's sunny outside? Hopefully you have experienced both sun and warmth and can agree with the statement that *on average* when the amount of sunlight is "high" we expect the temperature to be "high". How then can we make this quantitative?

One idea is to take our two quantities and compare them to their respective global means so that we have a quantity-dependent definition of "high" and "low." We then *multiply* the mean-subtracted quantities so that if both quantities are "high" then we get a large positive product. Similarly if both quantities are "low" at the same time, then we also get a large positive product. If one is "low" and one is "high" then we get a negative number. If we average all these products, then we can get a measurement that should correspond to our intuitive notion of correlation. The measurement we've just described is actually the formal definition of **covariance**, and is given symbolically in Equation 4.

At this point, you should know (but we'll point out) that your ability to estimate the covariance will depend on how much data you have. How will it depend? Well, the calculation of the covariance involves calculating the *means*, so you can probably make a good guess (or you can simulate it yourself!). More concretely, if the data is too small, then the sample means will be further from the true distribution more often, so that your estimation of covariance can be severely impacted. As always, you can now generate confidence intervals for your estimates with bootstrapping, and we encourage you to think about this, especially when the number of samples is small ($N < 50$).

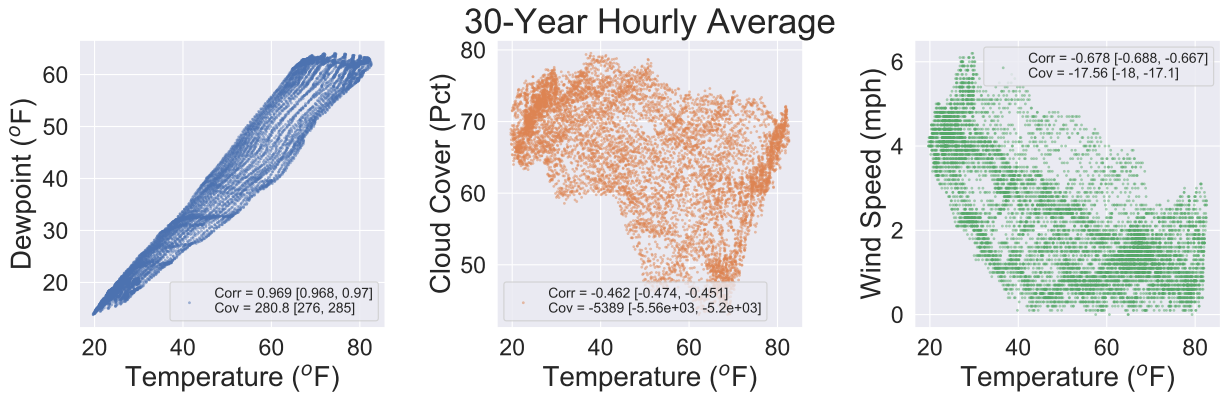


Figure 4: Weather measurements at O'Hare International Airport averaged over the last 30 years on an hourly-basis. The correlation and covariances for each pair of measurements are calculated along with bootstrapped 95% confidence intervals.

As an example of what this might look like with real weather data, consider Figure 4 where we've plotted a few measurements from O'Hare International Airport. It turns out that cloud cover and temperature are actually correlated in the way we intuited: when there are more clouds, it tends to be cooler. Or, quantitatively, we say that the covariance between cloud cover and temperature is -5389 [-5560, -5200]. However, without knowing the *scale* of these measurements, it's hard to interpret -5389 beyond the sign of the measurement. That is, we know the two quantities are inversely related because of the negative sign, but is -5389 a strong or weak relationship? As a comparison, consider the left panel of Figure 4, where the dew point and temperature are plotted against each other. By eye, this relationship seems much stronger than that between cloud cover and temperature, but it has a covariance of 280, which is 20 times smaller. So, to make this measurement comparable across data sets, we can account for the different scales of the measurements by dividing by the width of their distributions, as approximated with the standard deviation. This is called the **correlation coefficient** and is defined formally in Equation 5.

The correlation coefficient is useful because it gives us a standardized measure of correlation that is comparable across data sets. You should consider Equation 5 and convince yourself that the value of the coefficient will be between -1 and 1, where 1 is perfectly linearly aligned (what is the correlation of a variable with itself?) and -1 is perfectly anti-correlated. Aside from the interpretability issue, the covariance also is a less-than-ideal metric because simply changing the *units* of our measurement will change the value of the covariance. (Try this yourself by converting the degrees Fahrenheit into Celsius or Kelvin and recomputing the covariances.) The correlation coefficient also overcomes this deficiency. However, you should note that because the correlation coefficient involves computing a second-order moment, it will require *even more* data to make a good estimate.

Correlation and Covariance

The covariance between two quantities is defined as

$$Cov[X,Y] = \sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y}), \tag{4}$$

so that if two quantities always vary above and below their mean at the same index, i, then they will have a larger positive covariance, and if they oppositely vary, they will have a large negative covariance. If we normalize the terms of the sum by the standard deviations of each variable, we get the correlation between X and Y.

$$Corr[X,Y] = \sum_{i=1}^{N}\left(\frac{x_i - \bar{x}}{\sigma_x}\right)\left(\frac{y_i - \bar{y}}{\sigma_y}\right) \tag{5}$$

This normalization ensures that the correlation is confined to be between -1 and 1.

Now that we've made these definitions, we can build on them to motivate the ideas behind PCA. In particular, PCA will involve calculations using the **covariance matrix**, which is a matrix containing the covariances of each pair of measurements in a data set. That is, if we have X is a data set of N samples of K different measurements, then we can calculate the covariance between each pair of measurements. The covariance matrix arranges these so that the covariance between the i^{th} measurement and the j^{th} measurement is stored at the i^{th} row and j^{th} column of the covariance matrix. We'll discuss this more below, but for now, consider Equations 4 and 5 and try and understand why they correspond to an intuitive and simple notion of relatedness.

Try It Yourself

Download the hourly weather data from O'Hare here. Compute the covariance and correlation coefficients for different combinations of measurements. Use bootstrapping to estimate confidence intervals.

3.2 Principal Component Analysis

Consider the data set in Figure 5 where we have measured two quantities X_1 and X_2 about $N = 40$ times. We can estimate the means and variances of each X_i, but when they are plotted against each other as in Figure 5, we see that the two observations are somewhat correlated with one another. Indeed, using the formulas below, we can see that the two observations X_1 and X_2 are indeed correlated with a coefficient $\sim$0.5, meaning that they are not entirely dependent on each other, but they are not distributed independently either.

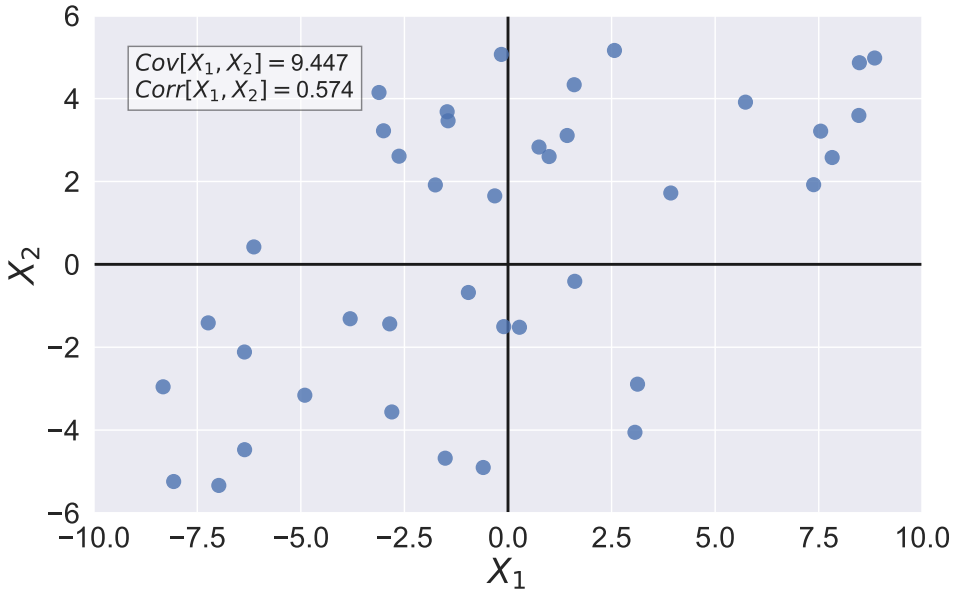


Figure 5: Illustration of correlated data.

As noted in the introduction to this section, we're concerned with finding (smaller, sparser, more interpretable) lower-dimensional descriptions of data. We've noted that sometimes (usually) measurements are correlated, so we want to know if we can leverage that to achieve this goal. Said another way, if our data are positively correlated, then knowing that X_1 is large means that X_2 is likely large as well, so being told about X_2 after being told about X_1 is *less-informative* than being told about X_1 in the first place. In this way, is there

a way to indicate where a data point is that takes the most advantage of the covariances in the data? The answer is yes, but to do this is going to take a bit of work.

Specifically, we're going to pose the problem as that of finding some *linear combination* of our covariates that captures most of the variation in our data set. We limit ourselves to finding a linear combination as a first pass at the problem and so that we can make use of the tools of linear algebra, but it's worth emphasizing that this is an assumption. In particular, as we noted earlier, linear models are (one of) the *simplest* types of models, and thus do the most to get around problems caused by low sampling density. Furthermore, many of the more complex algorithms that you might encounter continue to exploit this assumption, so it's worth paying attention to how it plays out here.

A Primer on Linear Combinations

A **linear combination** of quantities is a quantity generated by summing the product of each quantity with a constant coefficient. That is, if we have quantities $x_1, x_2, \ldots, x_K$ then

$$a_1x_1 + a_2x_2 + \cdots + a_Kx_K$$

is a linear combination of these x_i.

This idea can be generalized to *vectors* so that all plotted data points can be viewed as linear combinations. Specifically, if we have a point $\vec{x} = (a, b)$, we can think of this as being a linear combination of the **standard basis vectors** $\vec{e}_1 = (1, 0)$ and $\vec{e}_2 = (0, 1)$ (these are the vectors that point along the coordinate axes). That is, $\vec{x} = a\vec{e}_1 + b\vec{e}_2$.

However, we aren't confined to using the standard basis vectors to represent our data point. For example, consider $\vec{v}_1 = (1, 1)$ and $\vec{v}_2 = (1, -1)$, then $\vec{x} = \frac{a+b}{2} \cdot \vec{v}_1 + \frac{a-b}{2} \cdot \vec{v}_2$. (Set a and b to some values, say $a = 2$ and $b = 3$ for example, and confirm this works!)

If you need more of an introduction to vectors and linear algebra, these notes or these notes are excellent resources.

Put more concretely, we want to find the vector in data-space along which most of the variation in the data set occurs. In Figure 5, we can guess that this will be some line going from the bottom left to the top right. To visualize this, consider the animation in Figure **??** (You can download the gif here if your PDF viewer does not support embedded video). In

the animation, a data set similar to that in Figure 5 is shown, and a linear combination of X_1 and X_2 are shown as the orange axis through the data. As we explore all possible linear combinations by rotating this axis, the variance off the axis (denoted as "mean residual" in the inset plot) will vary, and we can find the direction that minimizes this off-axis variance (denoted by the green lines). We then assert that reporting the amount that each data point is along this new axis is a more "natural" descriptor of the entire data than one covariate at a time.

Now that we've tried to motivate the problem, we can tell you that finding this axis that minimizes off-axis variance is exactly the canonical dimensionality reduction method known as **Principal Component Analysis**, or PCA. PCA is predicated on analyzing the **covariance matrix** of a data set to determine a set of more "natural" coordinates, or **principal components**, along which to describe our data. The animation suggests that this can be thought of as a rotation, and as we'll see shortly, this can also be thought of as fitting a K-dimensional ellipsoid to our data, where the axes of this ellipse are a potentially more useful set of directions along which to look at our data.

Before we go any further, we're going to show an example of what this looks like. Consider Figure 6, where the data from Figure 5 is shown again, but with its principal components plotted as well. In this figure, the principal components have been found (we'll explain how in a moment), and have been indicated as orange and green lines. Then, the data have been *rotated* so that the principal components are the coordinate axes, and the directions corresponding to the original measurements are shown as the black lines. The effect of this on the shape of the data is shown in the panels on the bottom row, where we can see that we went from having a good amount of variance in both measurements (67% and 33% of the variance) to having most of the variance in the first principal component (82% and 18%). The idea then is that if we could only report *one number* for each data point (if we were reducing the dimensionality from 2 to 1, for example), reporting the data's position along the first principal component would capture most of the information. It's worth noting that while this example might seem trivial since our data are 2D, everything in this section generalizes to K-dimensions, so you can imagine that we're going to use this process to go from $K = 10$ million dimensions down to 3, for example.

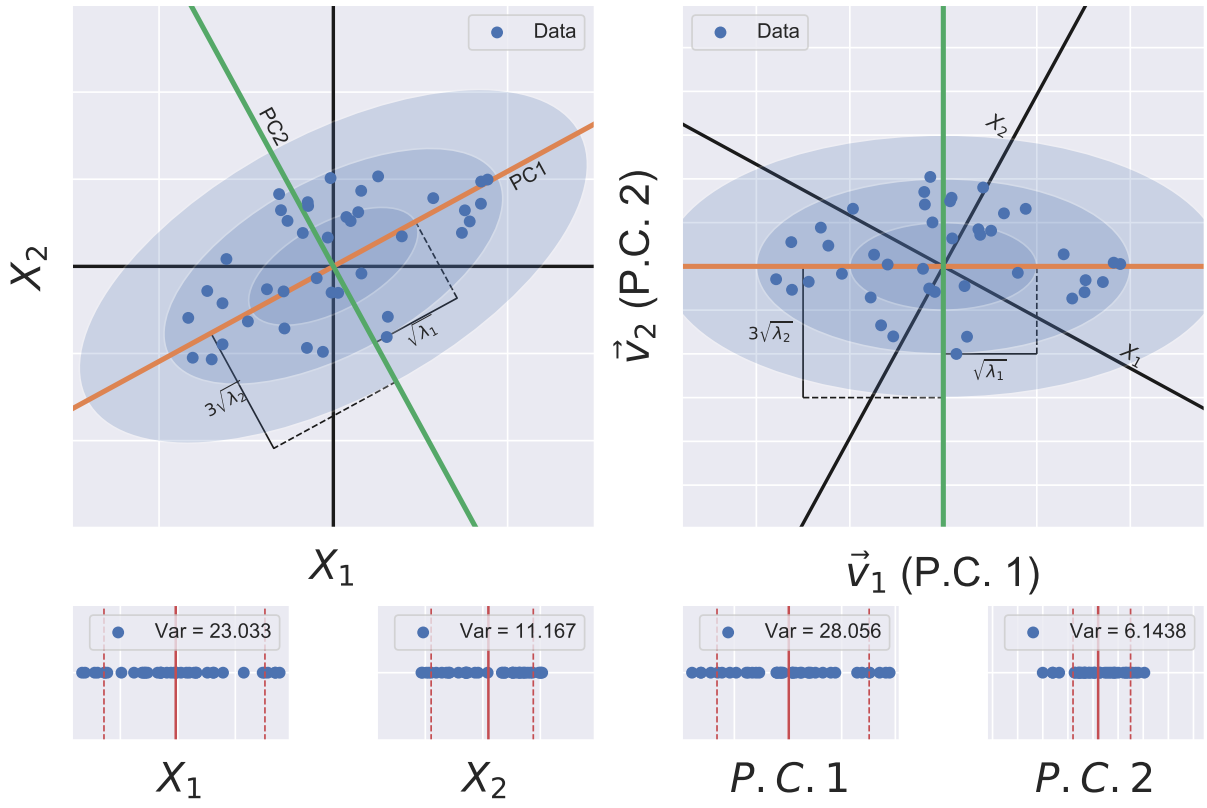


Figure 6: Data from Figure 5 is plotted in the original coordinates (top-left) and in principal component space (top-right). The first and second principal components are shown as orange and green lines, respectively. Ellipses fitted using the eigenvalues are shown behind the data in blue, where the scale of the ellipses is shown in units of the eigenvalues, λ_1 and λ_2. The bottom panels show the distribution of each coordinate in each space. The bottom left and center-left panels show the distribution of the X_1 and X_2 coordinates, respectively. The center-right and right panels on the bottom show the data's projections on to the first and second principal components, respectively. The solid red lines indicate the mean and the dashed red lines give the 10^{th} and 90^{th} percentiles. The *variance* of each quantity is indicated.

Big Idea of PCA

Figure 6 illustrates the goal of PCA. We want to find the transformation that moves as much of the variation in the data into as few directions (principal components) as possible. Using a subset of these maximally varying directions, we can (hopefully) reduce the dimensionality of the data set while (hopefully) capturing most of its features.

To describe how this works in more detail, first note that if we have N samples of K observations $\vec{x}_n = [x_{n,1}, x_{n,2}, \ldots, x_{n,K}]$ for $n = 1, \ldots, N$, then we can create a covariance *matrix* where the $(i, j)^{\text{th}}$ entry is the covariance between the i^{th} and j^{th} observations across all the n samples. If we center our data so that the mean of each column is 0, then we can calculate the covariance matrix using the matrix product $X^T X$ (`np.dot(X.T, X)`). Alternatively, in Python, the covariance can be calculated using `np.cov`, while the correlation coefficient is `np.corrcoef`.

For the data set in Figure 5, the covariance and correlation matrices, Σ and Σ_{corr} are

$$\Sigma = \begin{bmatrix} 23.624 & 9.447 \\ 9.447 & 11.453 \end{bmatrix} \qquad \Sigma_{corr} = \begin{bmatrix} 1.0 & 0.574 \\ 0.574 & 1.0 \end{bmatrix},$$

where the covariance or correlation between X_1 and X_2 is in the first row, second column of the respective matrix. You can convince yourself that the diagonal elements of the covariance matrix will be the respective *variances* of each of the observations and that the both matrices should be symmetric about their diagonals as a rule.

So how then can we use this to find *directions* that give directions of maximal variance and minimal off-axis variance (the principal components illustrated in Figure 6)? To do this, we'll need to use some mathematics, specifically, we'll use a technique from linear algebra called **eigendecomposition**. For those of you who have heard of this before, we'll cut to the chase: the principal components are the **eigenvectors** of the covariance matrix. As Figure 6 shows, the **eigenvalues** give the lengths of the axes of the fitted ellipse. If you have never heard of eigendecomposition, we will give a brief overview, but you should refer to an introductory text on linear algebra for more details.

The basic idea rests on the notion that all matrices are *transformations* in that if we apply them to an arbitrary vector with matrix multiplication, we transform the vector in a predictable manner. As an example, consider our covariance matrix Σ. If we apply Σ to the

unit vector in the x-direction, $\vec{e}_1 = [1, 0]^T$, then the result is the vector $\vec{u}_1 = [23.6, 9.4]^T$. That is, Σ *transforms* the x-axis into a line going through $(23.6, 9.4)$. Similarly, Σ transforms the y-axis into $\vec{u}_2 = [9.4, 11.5]^T$. We can see this graphically in Figure 7, where some interesting vectors have been transformed with Σ. In this figure we have highlighted the transformed coordinate vectors $\vec{u}_1$ and $\vec{u}_2$ as well as two other vectors of interest, which we've labeled $\vec{v}_1$ and $\vec{v}_2$.

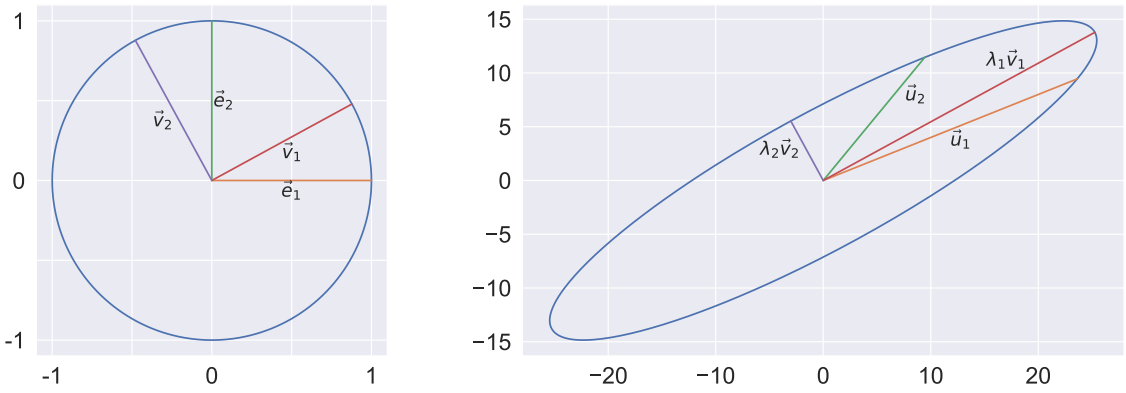


Figure 7: Illustration of the linear transformation Σ as applied to the unit circle. The coordinate unit vectors $\vec{e}_1$ and $\vec{e}_2$ are transformed into $\vec{u}_1$ and $\vec{u}_2$ as described in the text. They are indicated by orange and green lines, respectively. The eigenvectors of Σ are also indicated by red and purple lines and are denoted $\vec{v}_1$ and $\vec{v}_2$.

The vectors $\vec{v}_1$ and $\vec{v}_2$ are interesting because unlike the other vectors on the unit circle, they aren't twisted to a new direction by Σ, they are only shrunk or stretched (in this case stretched). That is, these vectors sort of get mapped onto themselves by Σ. It turns out that every symmetric $K \times K$ matrix has K of these vectors and they are special enough to have earned the name of **eigen**vectors (German for "proper" vectors). The amounts by which these vectors are stretched or shrunk are called the **eigenvalues**. More concretely, the eigenvectors and eigenvalues are the vectors, $\vec{v}$, and values, λ, that satisfy the equation

$$A\vec{v} = \lambda\vec{v} \tag{6}$$

for some matrix A.

So why then do we want to find the eigenvalues and eigenvectors of the covariance matrix? What does this have to do with the problem we set out to solve? We can see why this make sense in a few ways, but first it's worth noting that when we applied Σ to the unit circle, we generated an ellipse. Furthermore, the axes of this ellipse were the eigenvectors. This outcome is entirely generic for any matrix, so for our covariance matrix generated from

data, the eigenvectors give the longest and shortest vectors on the covariance ellipse. So if our goal is to find the direction with largest on-axis variance and smallest off-axis variance, then this direction is naturally found by the eigenvectors. In thinking about the covariance this way, these ellipses are sometimes called **confidence ellipses** because they correspond to the higher-dimensional *confidence intervals* of an estimate of a multi-dimensional mean. These confidence ellipses are indicated on Figure 6.

To see this another way, we note that when we calculate the covariance matrix and use it to describe our data, we're approximating it as a Gaussian distribution. This is because Gaussian distributions are *the* distributions that minimize entropy when we know the first two moments of our data. That is, if we represent our data with a mean and variance, then we're representing it as a Gaussian distribution, which you might be surprised to learn is an *ellipsoid* in more than one dimension! Again then, finding the axes of the ellipsoid will give us the directions of greatest variation, by definition.

In addition, the eigenvalues give the *variance* of the data along the principal components and the sum of the eigenvalues gives the "total variance" of the data. In this way, dividing each of the eigenvalues by the sum of all the eigenvalues yields the **proportion of variance explained** by each principal component. The process of eigendecomposition doesn't inherently order the eigenvalues and eigenvectors, but most algorithms return them ordered by their eigenvalues, so that it is convention that the first principle component corresponds to the most variation, the second PC corresponds to the direction of second-most variation, and so on. In the example in Figure 6, the first principal component explains 82% of the variation in the data, and since the data is 2D, the second PC explains the remaining 18%. As we'll discuss later, it is not necessarily a given that there will be such a discrepancy in the variance explained by the different PCs.

More practically, once we have the eigenvectors, we can use them as new coordinates for our data. So the coordinates of a vector $\vec{x} = [a, b]^T$ in the new space are found by calculating the **projection** of the vector onto each eigenvector. A projection can be found by computing an inner product (dot product) between the two vectors (`np.dot` in Python). The eigenvectors[2] for Σ are $\vec{v}_1 = [0.878, 0.479]^T$ and $\vec{v}_2 = [-0.479, 0.878]^T$, so our new vector would become

$$\vec{x}^* = \begin{bmatrix} \vec{x}^T \cdot \vec{v}_1 \\ \vec{x}^T \cdot \vec{v}_2 \end{bmatrix} = \begin{bmatrix} 0.878a & + & 0.479b \\ - \; 0.479a & + & 0.878b \end{bmatrix}.$$

That is, in principal component space, $\vec{e}_1$ becomes $[0.878, -0.479]^T$ and $\vec{x}_{example} = [3, -2]^T$ becomes $[1.676, -3.192]^T$. To interpret this a bit further, what we're saying is that the vector

[2]Note that by convention, eigenvectors are typically *normalized* so that their length is 1.

that was $[1, 0]^T$ in the original X_1-X_2 measurements, has 0.878 units of PC1 and -0.479 units of PC2. Similarly, the vector that was $[3, -2]^T$ in the original X_1-X_2 measurements, has 1.676 units of PC1 and -3.192 units of PC2. Thinking about coordinates as "units of axes" this transformation moves from "units of measurements" to "units of principal components". In Figure 6, this is the difference between the left and right panels: in the left panel, the bottom-axis-coordinate of each point is the original data measurement X_1 and in the right panel, the bottom-axis-coordinate is each data point's distance along the first principal component. If you're still a little worried about this, don't worry, we're going to give you some code to help you build this up on your own as well.

Again, we only want to give you some intuition about why it makes sense to think about covariance matrices and their eigenvectors, not delve too deeply into related mathematics. You should at a minimum take away that covariances (or matrices more generally) correspond to ellipses and that we want to get the axes of these ellipses. Eigendecomposition gives us these axes. In the following subsection we'll discuss how you should *use* this new idea.

3.2.1 How to Use PCA

Ok, so we spent a good amount of time talking about the why of PCA, let's remind ourselves why we're going to this trouble. First and foremost, in a situation with limited information, approximating our data with the first two moments and examining that approximation is not a bad idea. In this way, calculating the covariance matrix and examining it's eigenvectors and eigenvalues can be useful. But the reason that we include PCA as the first example of dimensionality reduction is because of the promise of capturing most of the information in our data with a just a few coordinates. In this way most of the discussion about PCA centers on the question of "how many PCs to use?" That is, as we noted above, PCA effectively amounts to finding a special rotation of our data (once we've centered on the mean) – it doesn't reduce the number of axes.

In practice then, there are two main heuristics for deciding how to reduce the number of PCs. The first is to simply take as many PCs as you need to explain some fraction of the variation in your data. A common threshold is 80%, where the hope is that 80% of the variation corresponds to only a few PCs and that you're setting an upper bound on the amount of potential information that you're throwing away. However, at this point you should be skeptical that there is a magical threshold that works for all data, and indeed, concern about this heuristic usually results in a referral to the second heuristic, which is known as the "Elbow Rule."

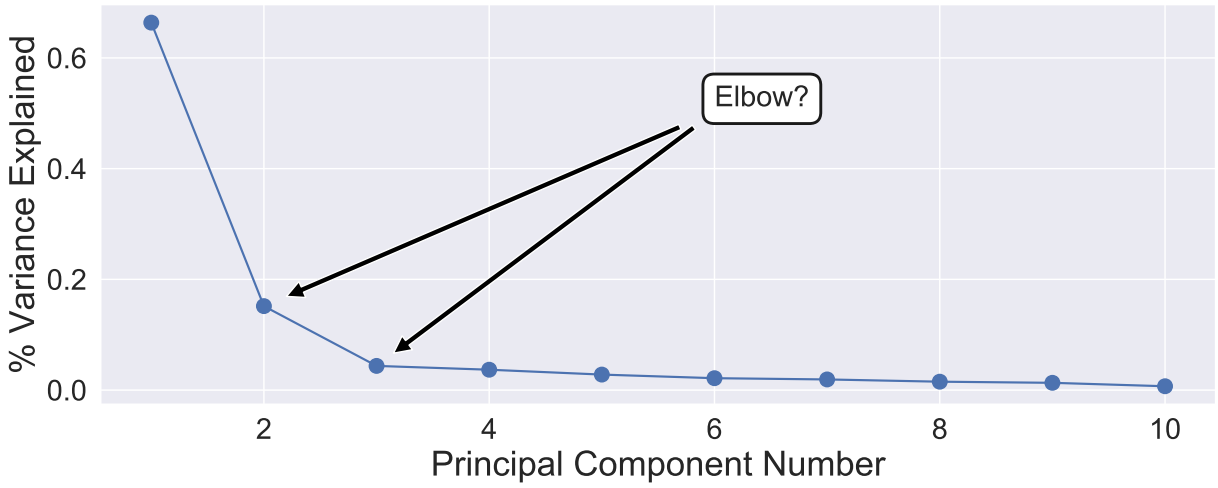


Figure 8: Scree plot of eigenvalues of covariance matrix from data from Figure 5 that have been embedded in 10 dimensions. The potential "elbows" in the plot are indicated.

To illustrate the "Elbow Rule" we introduce the **scree plot**, which is a plot of the ordered eigenvalues (or the % variance explained) in descending order. The "Elbow Rule" then asserts that you should look for the "elbow" in the plot, as this indicates where the "meaningful" PCs transition into describing only noise. In Figure 8 we show the scree plot generated by adding 8 axes of Gaussian noise to the data from Figure 5 and applying PCA. As expected, the first PC explains most of the variance in the data, albeit less than the 82% from earlier. Then since the data really are 2D, the second PC also has some explained variance, and the subsequent PCs all individually explain less than 5% of the variance in the data. Then, depending on your interpretation of the "Elbow Rule", we would truncate our data to only being expressed with the first two or three principal components.

Note that while the thresholding heuristic at least bounded the variation that we're eliminating, the "Elbow Rule" doesn't guarantee any particular amount of explained variance. However, this rule is also suspect in application for two reasons: we rarely have enough data to see the elbow, and data are rarely *linear*, as we've assumed.

To address the second point first, consider the helix in Figure 9. In this example, the data actually are one-dimensional, but non-linear, so that you need all three principal components to capture at least 80% of the variance in the data. Furthermore, the scree plot doesn't really indicate an elbow, so that heuristic also fails to reduce the dimensionality of the data. Now, this example might seem a bit pathological, but you should get a piece of paper and convince yourself that you can't accurately describe a circle or a parabola with

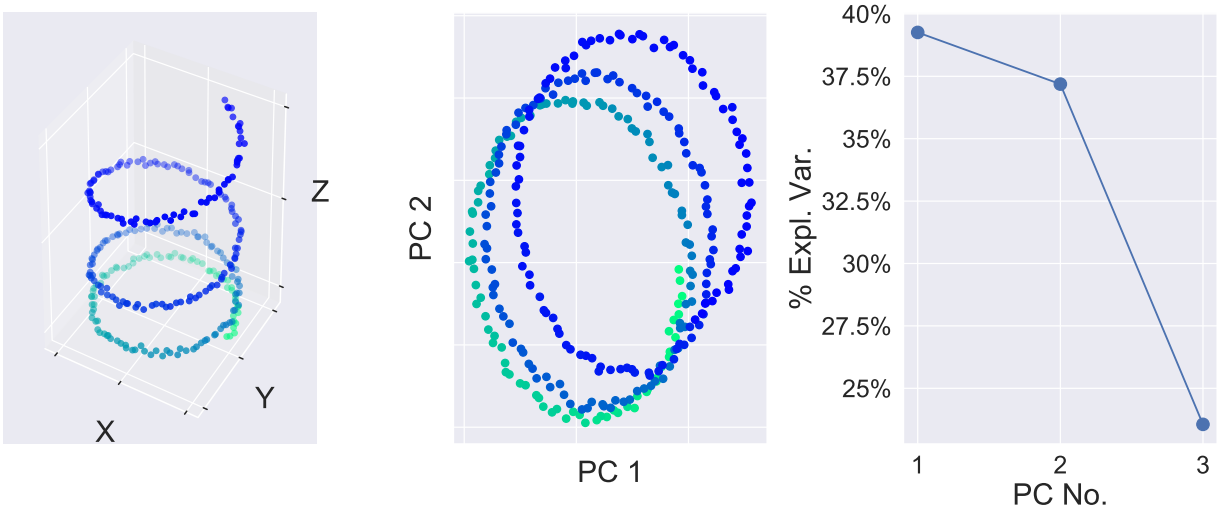


Figure 9: Illustration of a helix in 3D and the scree plot of its PCA eigenvalues. The center panel shows the projections onto the first 2 eigenvalues.

a single straight line. Any data set that has any such shaped wrinkle will result in a PCA scree plot that lacks a clearly defined elbow.

As for our concerns about having enough data, at this point we should be able to point at the Central Limit Theorem and the Curse of Dimensionality to convince you that estimating a complicated quantity such as an eigenvalue will require decent sampling density. However, you now know how to circumvent some of these effects. For example, if you were going to apply some 80% threshold to select PCs, you could at least bootstrap to use the number of PCs that gives you 80% explained variance 95% of the time! However, as we'll explain shortly, we can use our methods developed in the previous module to address all these concerns.

Before we move on, it's worth noting that while it might seem that we're focusing on the deficiencies of an algorithm (PCA) before we've even finished telling you how to properly execute it. This is by design in that we want to make critical thinking a front-and-center part of your journey towards learning about your data. However, you should not take away from this that we've introduced one bad algorithm and the good ones are all saved for another class. It's quite the opposite: PCA is probably the most robust and interpretable method out there, and all of these concerns apply to all machine learning techniques.

In this context then, let's see how we can use the abilities of our computer to make the best assessment that we can. Specifically, the idea that we'll illustrate is that we can use **marginal resampling** to generate a **null distribution** for the eigenvalues in our data. Principal components whose eigenvalues seem to be consistent with noise can be ignored,

and we can persist by using just the most statistically non-random principal components.

The MNIST Digits

To illustrate the synthesis of all of the components of this course into a statistically-grounded PCA we're going to introduce the MNIST Digits data set. This data set consists of thousands of hand-written digits that were collected by the U.S. Federal government from Census employees. These digits were converted to 28×28 grayscale pixel images. The full data set contains some 70,000 images, which is somewhat large, so we're going to use a reduced data set here. The correct digit label for each image is then given here. The images we've provided have also been converted from grayscale to binary, so that each pixel is either a 0 or a 1. An example of each type of digit is shown in Figure 10.

Now, you may be wondering how *images* fit our description of an $N \times K$ data set of N samples of K observations. The answer is that an image is just an arranged list of pixel intensities, so in this case, we can think of a 28×28 image as a sample of 784 pixel measurements! Indeed, when you load in the data, (using `np.loadtxt`, for example), you will see that the shape of the array is `(2500, 784)` and to show an image using `plt.imshow`, you'll need to `reshape` it into a 28×28 array.

The idea is that we can "unwrap" images, apply PCA, and find directions in "pixel-space" that correspond to the most variation in the images. Then, using just these directions, we might be able to represent the images without using 784 numbers – ideally just a few!

Figure 10: Example of each digit from the MNIST digits data set.

You'll explore this more thoroughly in a worksheet, but in these notes we'll illustrate what this looks like. First, let's perform PCA and take a look at the different parts. Our heuristics suggest that we can look at the scree plot (ordering of eigenvalues) to deduce the number of eigenvectors we'll need. This is shown in Figure 11.

In this Figure you can note that there is a relatively smooth spectrum of eigenvalues, not the sharp elbow as suggested by the "elbow" heuristic. However, if you had to, you could maybe say that there is a change in behavior somewhere between the 10th and 25th principal component based on the left panel. However, the center panel suggests that this might only account for 50-70% of the variation in the data!

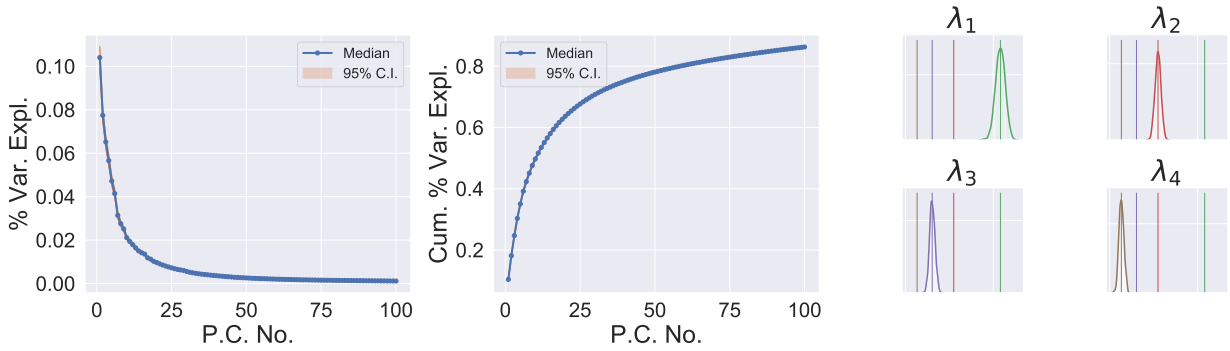


Figure 11: Bootstrapped scree plot for the MNIST data set. Median bootstrapped values and 95% confidence intervals are shown in the left and center panels for the first 100 eigenvalues (confidence intervals may not be visible because they are very small). The right set of panels shows the distributions of the first 4 eigenvalues. Their medians are shown on all 4 plots by color-coded vertical lines.

Let's put this aside and check to see what the principal components are even picking up. Figure 12 suggests that there *is* some structure to the data (although in the worksheet you will have the opportunity to come to your own conclusions). In particular, we might say that the first principal component is differentiating between the 1's and the 0's. That is, PCA is suggesting that if you could only use one piece of information to describe where an image is in digit-space, you would be best off knowing how 1-like or how 0-like your image is. Indeed, considering Figure 13, we see that the first principal component looks like a bright 1 and a dark 0 on top of each other. Thus, images that look like zeros will be positively projected onto this image, while images that look like ones will be negatively projected (in Figure 10, the black pixels are zeros and the cream pixels are ones!).

So, since we can see the labels and we know something about digits, it seems that this PCA embedding might have been useful! The other digits are still a bit jumbled, but the

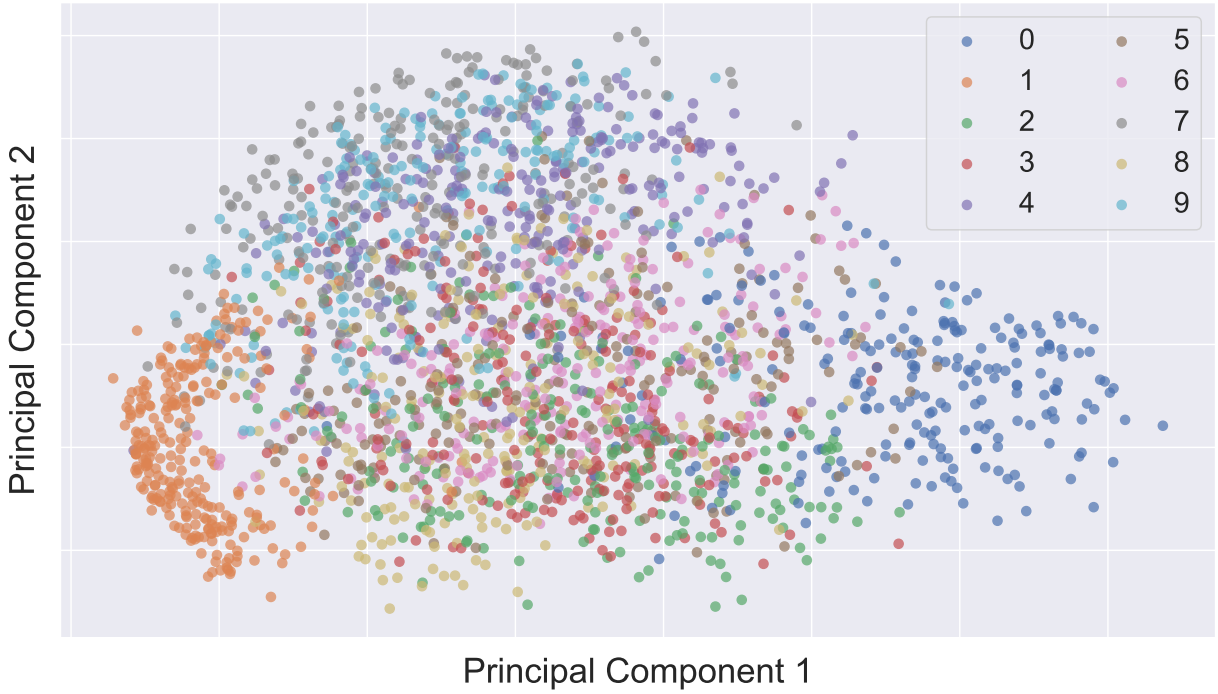


Figure 12: Projection of data onto the second PC vs the projection onto the first. Digits are indicated with differently colored dots.

other PCs might be useful to separate them until we can see the complete structure of the data. Indeed, looking at Figure 12, we can infer that the second principal component has something to do with the difference between 2's, 3's, 6's, and 8's as compared to 4's, 7's and 9's. During the worksheet, you should spend a moment examining the second principal component to see if this makes sense to you.

So how then can we determine how many of these potentially very informative components we should use? The answer comes from thinking statistically and quantitatively about what the "elbow rule" was trying to identify: we want to know when the components transition from telling us about well-sampled directions in the data to those that are just noise. We now know how to use simulation to generate null hypotheses, so let's simulate the application of PCA to data that has no structure to it. That is, we can generate fake MNIST images that have no covariance - the knowledge of one pixel tells me nothing about its neighbors. An example is shown in Figure 14

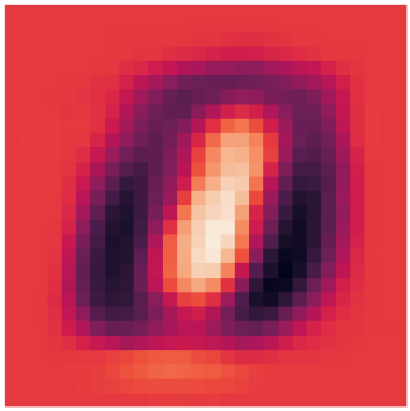


Figure 13: First P.C. of MNIST data. Lighter pixels indicate positive **loadings**, darker indicates negative.

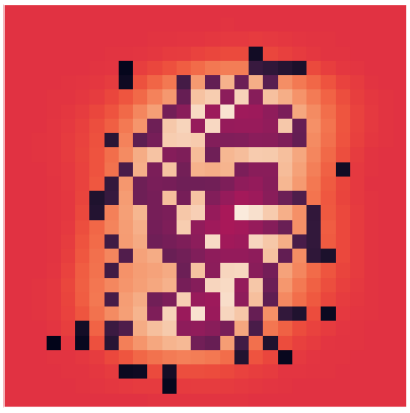


Figure 14: Example of an image that has been resampled from the marginals of the original data.

In Figure 14, each pixel was drawn randomly from the pixels at that location in all the (mean-subtracted) images. As a result, there are no dark spots around the edges, because almost none of the digits have such marks, so it would be unlikely to find a point in that space. However, as we get closer to the middle of the image, the likelihood of a particular pixel being "ink" is higher, so there are more dark spots. The idea is that such images will be reflective of the larger space in which the data lie, but without the correlations between pixels. Since PCA is trying to use correlations to find structure, a null hypothesis is that there is no true covariance and the eigenvectors that are found are all due to random noise.

This brings us to the final figure of this section, in which we generate a distribution of eigenvalues for each principal component after applying PCA to many null-shufflings of the data. (We sometimes call marginal resampling **shuffling**, because it is akin to "shuffling" each column of the data independently.) This can be seen in Figure 15.

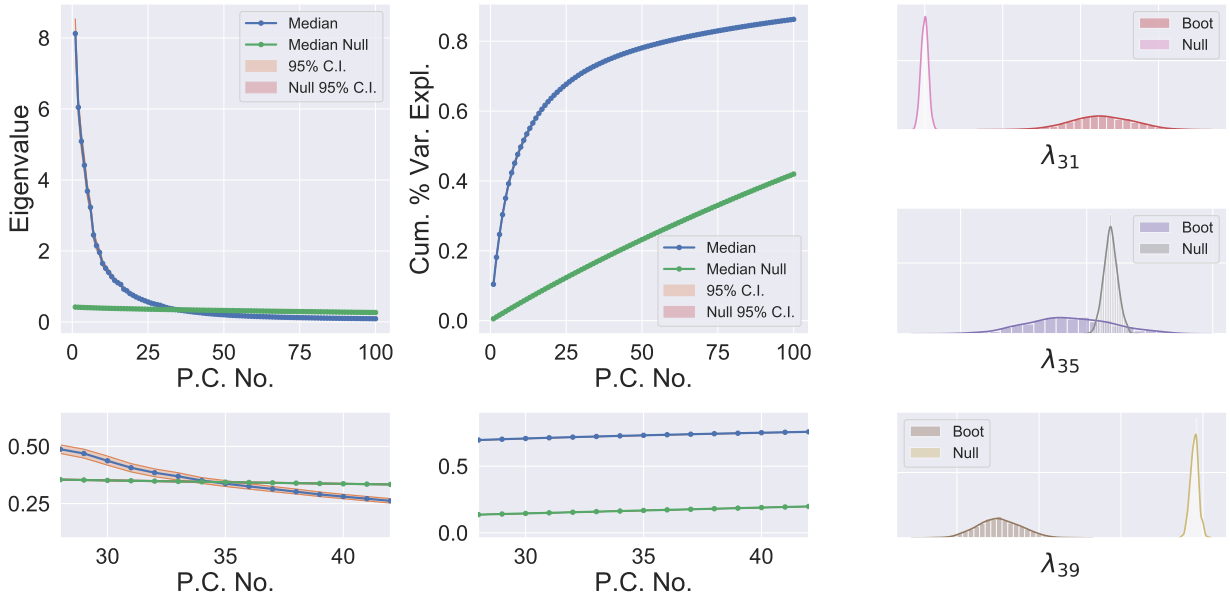


Figure 15: Scree plot of the bootstrapped and null eigenvalues. The bottom left and bottom center plots have zoomed in on PCs 28 through 42. The right 3 panels show the distributions of eigenvalues changing positions as the scree plots intersect.

Figure 15 shows that the eigenvalues for the shuffled null data are much smaller than those of the real data for about 35 PCs until the curves cross and the data's eigenvalues become smaller. As a result, you would be justified in keeping ∼34 principal components, but more or less than this might be throwing out data or including components below the noise floor.

You might be concerned however that because the different eigenvalues are themselves correlated – removing more variation in one direction necessarily reduces the size of the others – so making this comparison on a PC-by-PC basis might not be this simple. This concern is somewhat ameliorated by the fact that these are distributional comparisons, but we can also be a bit more conservative, as shown in Figure 16. In this figure, we compare *all* of the null eigenvalues with *all* of the data's eigenvalues and only use PCs with eigenvalues that are above the entire noise distribution. Based on this figure, it might seem that the 30th and 31st principal components are right on the edge of component sizes that were seen in null data, but that those with larger eigenvalues are definitely not consistent with noise.

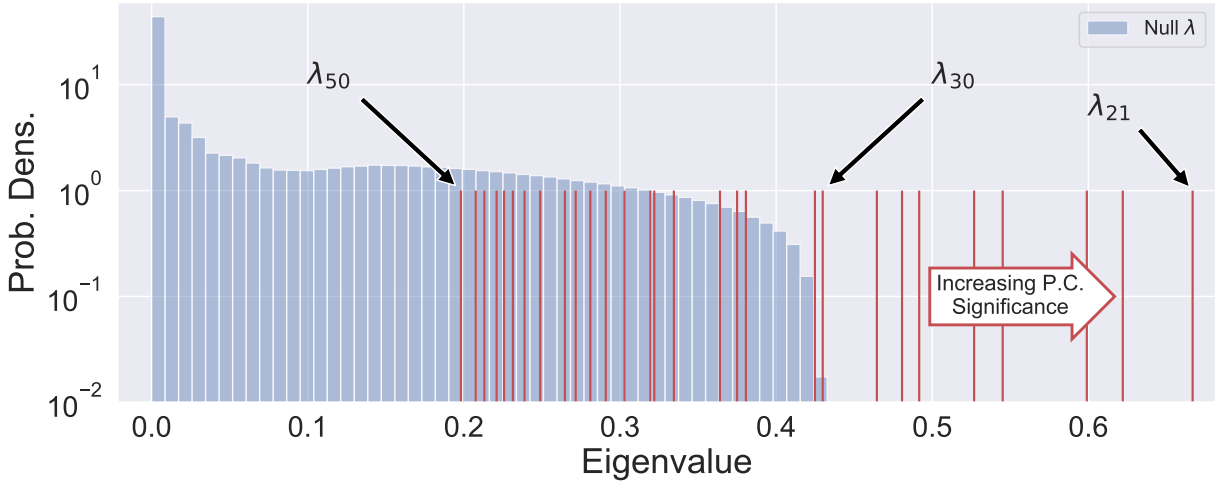


Figure 16: Distribution of eigenvalues resulting from applying PCA to 1000 marginal resamplings of the MNIST data are shown in blue. The first 50 eigenvalues from the data are shown as red lines (some not shown for visualization purposes). A few interesting eigenvalues are annotated. Notice the log-scaling of the vertical axis.

If you are still concerned about this, we of course have pointed out that you can use other metrics to decide "quality". For example, set up a training-testing regime that uses different numbers of PCs and calculates the "reconstruction error" between the real data and the re-represented data using only the PCs. If you are using PCA to identify covariates to use in a predictive model, use as many PCs as maximize prediction. While we have presented

this method to you as a relatively robust and quantitative method for using PCA to reduce dimensions, you should not stop thinking about how you can leverage your computer to solve *your problem* with *your data*.

At this point, we hope you see that there is a lot more to discuss about PCA than you might find in other places. In particular, PCA makes a lot of assumptions, but they are necessary to get a handle on a really difficult problem. Also, while the math of PCA is elegant and interesting, some theoretical aspects are rarely born out by real, sparse, and non-linear data. We also spend this effort both to get you used to the level of criticism that you should apply to all mathematical and computational methods, but also to illustrate how the tools we've introduced in the first few modules can be directly applied in helping you make these assessments. On this note, we'll now move on to LASSO, which is another landmark result in machine learning that deserves some scrutiny and appreciation.

How to Use PCA

Here are some things to do when performing PCA:

- Look for elbows in your scree plot. Bootstrap your data to assess the robustness of this feature.
- Shuffle your data and apply PCA. Compare the distribution of eigenvalues to those of your non-shuffled data.
- Make some scatter plots of the most significant PCs and examine the contributions of the measurements to each PC. Are the significant directions patterned in an unexpected way?

Try It Yourself

At this point you can try out Worksheet 4.1. In this worksheet you'll replicate the figures in this section and more. Attempting the worksheet is likely the only way you'll really start to understand this topic!

3.3 LASSO

The methods described in the previous section on PCA gave a route to using a statistical way of thinking to performed unsupervised dimensionality reduction more effectively. We have already introduced linear regression and the premise of over and under fitting in the context of simple polynomial and linear models. In these cases, the idea was that ordinary least-squares regression, given some covariates and a response variable, would produce some weight (a regression coefficient) to each covariate. Because there's nothing special about zero to OLS, and since we've given the parameters as wiggle room, these fitting techniques will give weights to those coefficients, *even if the underlying truth is zero.*

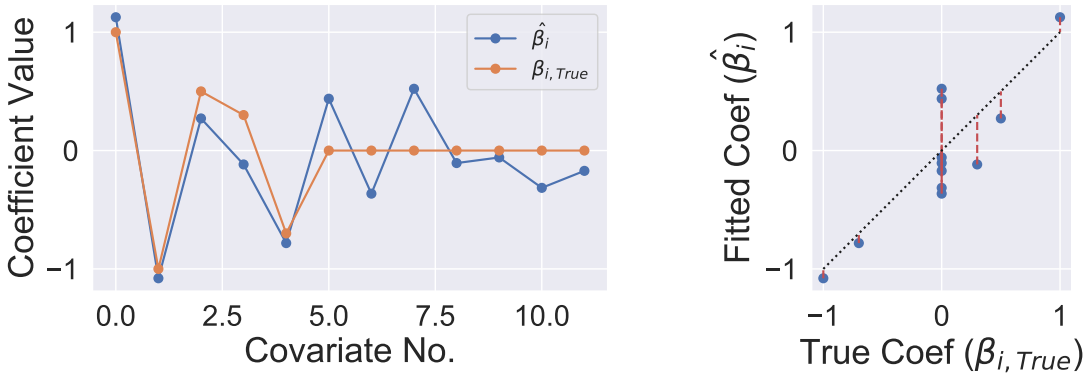


Figure 17: (Left) Fitted and true regression coefficients for each of the 11 coefficients. Plot of the fitted coefficients as a function of the true coefficients. Discrepancies are indicated by red dashed lines. The line corresponding to perfect matching is given in black.

As an example consider Figure 17, where some 11 covariates, X, were used to generate a response, Y (plus some noise). The first 5 covariates had non-zero coefficients, while the last 6 had no effect on Y. The data were then fit using a standard linear regression implementation and the results are indicated by the blue line and dots. In all cases, the predicted regression coefficient is not zero.

Now, you might be (rightly) concerned that this has more to do with poor sampling than the fitting routine being deficient, and indeed you are correct that we spent two modules discussing how you can detect the robustness of a fit. However, the example in Figure 17 is unrealistic in the way that we've only just started talking about: there are still not that many covariates. As we increase the covariates and reduce our sampling density the likelihood that many of our covariates will be non-negligibly zero will increase, but so will the likelihood that a coefficient that's actually zero seem to be a real predictor. In this context, and in the

context of dimensionality reduction more generally, we want to develop a method that sets $\hat{\beta} = 0$ with some frequency.

In a more practical sense, we really do want model selection algorithms that can systematically and predictably select between different models while optimizing different goals. At the beginning of these notes we compared the prediction error due to different models. It would be excellent to have a method that would use this to make a formal selection between them!

So then thinking about how our current model fitting works and thinking about what we'd like to happen, what inroads can we make? Consider the example in Figure 17, what if we just kept the largest coefficients? Then choosing any threshold that wouldn't eliminate real coefficients would also only eliminate the 9th covariate. As indicated in Figure 18, even using the confidence interval would result in eliminating covariates that actually are non-zero.

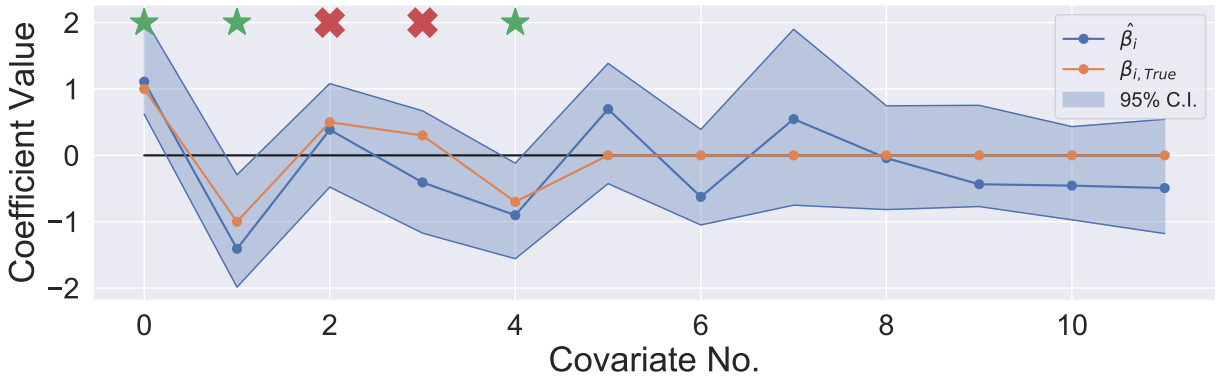


Figure 18: (Left) Fitted and true regression coefficients for each of the 12 coefficients. A bootstrapped 95% confidence interval is indicated by the shaded region. The stars and exes indicate the accuracy of using the confidence intervals to eliminate covariates.

In this way, the earlier discussion of model **capacity** was a prelude to our present discussion of **regularization**, which is a method that promises to adjust the capacity of a model in a way that balances tendencies to over- and under-fit. If such a method exists, it would be exactly what we're looking for, right? Well, as with all things, to get the method that we want, we're going to have to give something up. To understand how this method works and what we're hinting at, let's get into some details.

Consider a two variables X and Y (maybe from Figure 3) and we want to fit the model $Y = \beta_0 + \beta_1 X + \beta_2 X^2$. If I wanted to use linear regression I could simply relabel $X = X_1$ and $X_2 = X^2$ and the fitting method would be none the wiser. In this way, the general model

for linear regression is

$$y_i = \sum_{k=1}^{K} \beta_k x_{i,k} = \vec{x_i^T} \cdot \vec{\beta} \tag{7}$$

where $x_{i,k}$ is the k^{th} covariate of the i^{th} sample, and β_k is the k^{th} regression coefficient. We can get rid of the intercept term β_0 either by centering X or adding in X_0, which is a vector of all ones.

Recall then that the OLS approach to linear regression was to minimize the sum of the squared residuals. More formally:

$$\hat{\vec{\beta}} = \min_{\vec{\beta}} \left| Y - X\vec{\beta} \right|_2^2, \tag{8}$$

where $|\cdot|_2$ is the **Euclidean**-norm, or the standard vector length formula you might be familiar with, so $|\cdot|_2^2$ is the square of the 2-norm. The idea of **regularization** is to say, that's nice that we want to minimize the distance away from the data, but we also want to minimize the size of the regression coefficients. Specifically, we want to incentivize some of them to be *zero* while still maintaining some fidelity to the data. One way to write down this compromise is to instead write

$$\hat{\vec{\beta}} = \min_{\vec{\beta}} \left[\left| Y - X\vec{\beta} \right|_2^2 + \lambda \left| \vec{\beta} \right|_\alpha \right], \tag{9}$$

where now the new term is a *constraint* (λ is a Lagrange multiplier) on the size of the vector of regression coefficients. More intuitively, we've said that the method should fit the data as best it can, but it only has so much coefficient "material" to use. The amount of "material" available depends on the new parameter λ. When $\lambda = 0$, we're back at OLS, and when $\lambda \to \infty$, the method sets all the coefficients to 0. The hope is that there is an intermediate value of λ for your problem that balances these two extremes.

Now you might have been worried about the other variable in the problem above, α. This parameter is a bit subtle - it determines the **metric** that we use to calculate the size of $\vec{beta}$. If we set $\alpha = 2$, then we recover the Euclidean distance, that is $|\vec{\beta}|_2 = \sqrt{\sum_i \beta_i^2}$. When we use $\alpha = 2$ in our regularization problem, it's known as **Ridge Regression**[3]. Setting $\alpha = 2$ is something that was often used because it's very computationally well-behaved; specifically, it's a quadratic problem, which means that it can always be solved in finite time. However, it turns out that ridge regression doesn't work quite as well as we want; it still permits variables to remain close to zero but at zero too often.

[3] https://en.wikipedia.org/wiki/Tikhonov˙regularization

In this way, what we really want is to have the problem simply minimize the number of non-zero coefficients. This is actually what corresponds to setting $\alpha = 0$. Mathematically, when we write $|\vec{\beta}|_0$ (or some similar notation), we mean "how many non-zero elements are in $\vec{\beta}$? Unfortunately, this problem is computationally very difficult, and all known methods for solving it suffer from some deficiency that setting $\alpha = 2$ would solve. As a compromise, LASSO sets $\alpha = 1$.

LASSO is actually an acronym for **Least Absolute Selection and Shrinkage Operator**, which is a method proposed by Tibshirani in 1996, but you can think of it more intuitively as being a method that "lassos" some coefficients and brings them down to 0. While it might have seemed obvious to simply split the difference between $\alpha = 2$ and $\alpha = 0$, it wasn't actually obvious that $\alpha = 1$ would have enough of the properties of both, which it thankfully does.

In any case, discussing the details of optimizing the LASSO function or why it has good computational properties is beyond the scope of this course. What you should understand though is that LASSO is a method that attempts to balance model sparsity (potential to under-fit) with model capacity (potential to over-fit). Even more practically, you should be concerned that we have introduced a new variable into the problem, λ! We said earlier that depending on the value of lambda we'll either revert to OLS or suppress all the regression coefficients unnecessarily.

As an example of what this looks like, consider Figure 19. On the left side of this figure, all the regression coefficients have non-zero values. As the parameter λ is ramped up, more and more coefficients are set to zero, until eventually all are. In the data that is being shown in Figure 19, 100 samples of 200 covariates were given to the LASSO algorithm, and only 10 of the covariates were truly non-zero. While these covariates had coefficients that were generally larger than zero-covariates, not all of them did, and some of the non-zero covariates were even pruned to zero before zero-covariates, which you can see on the right side of the figure.

So how then do you choose what λ to use? Again, the answer is something that we've already discussed: **cross-validation**. Since λ is a *hyperparameter* (an algorithmic parameter) and not a quantity in the problem, CV is the most straightforward method for finding an optimal value.

Of course, you now know that you can take this one step further and generate synthetic data that are consistent with a null hypothesis that there *is no relationship* between the quantities of the problem. In that case, what does LASSO generate? It still characteristically sends all variables to zero when λ is large, but does prediction error dip for intermediate

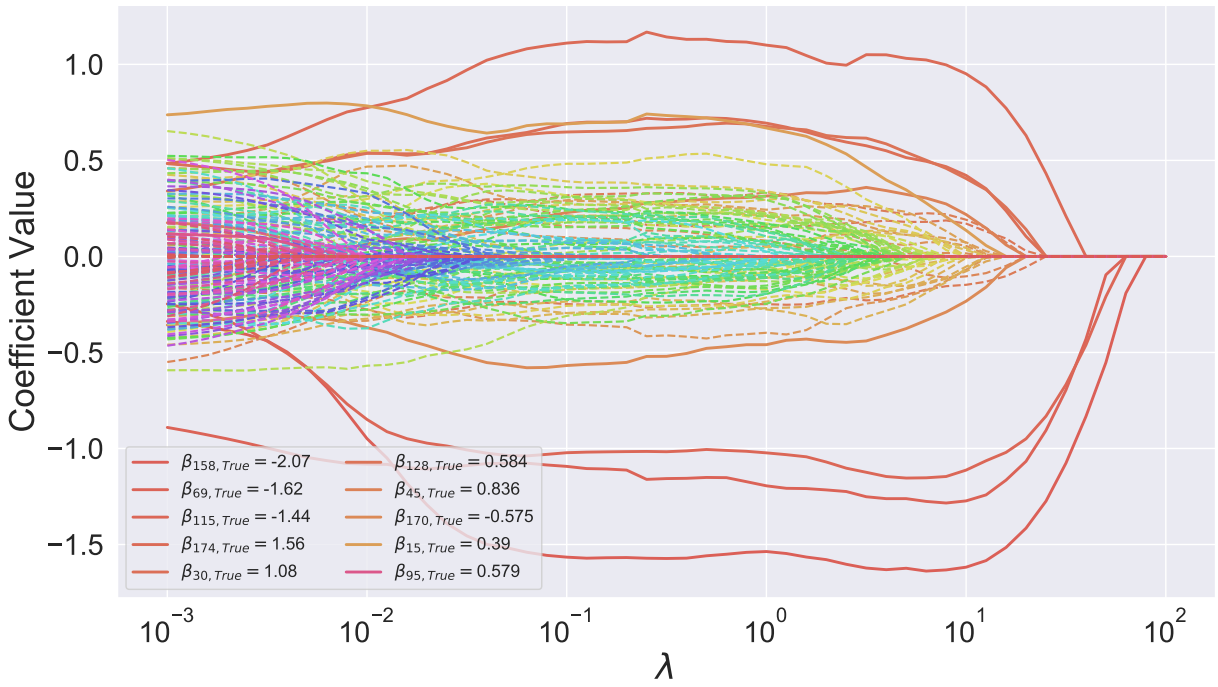


Figure 19: Sizes of the regression coefficients found by LASSO as a function of the parameter λ. The true data had 10 non-zero coefficients, which were indicated in the legend and by solid lines. Covariates that had coefficients that were actually zero are indicated by dashed lines.

values due to some peculiarities of your data? We show what all of this looks like in Figure 20.

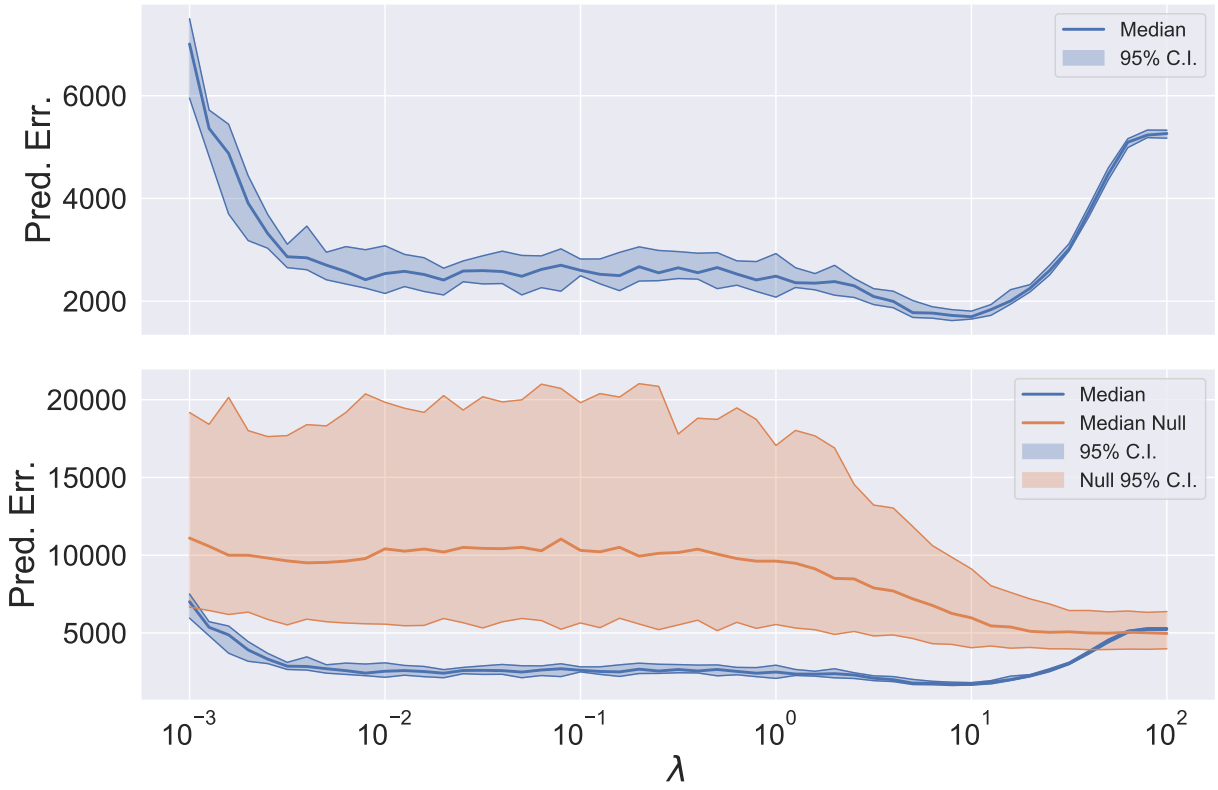


Figure 20: 10-Fold Prediction error and bootstrapped confidence intervals as a function of λ (`alpha`) from the data (blue) and null data (orange).

In this figure, we demonstrate the results of bootstrapping, null generation, and their synthesis, in the context of LASSO. First, off using the data from Figure 19, we can see in the top panel that there is indeed a minimum in the prediction error near $\lambda \approx 8$. What is interesting then is that the general trend in the errors in the null distribution also roughly follows that of the data, albeit without the minimum. In this way, we certainly see (and we know because we constructed the data) that the model is actually capturing a signal in the data.

Try It Yourself

Feel free to attempt Worksheet 4.2!

4 Chapter review

Fundamentally, the point of this chapter is simply to repeat all the content of the other modules, but in the context of a real problem. We hope you see that rather than being a quirky way to use your computer, this framework has something to say about algorithms and analyses that have been around for decades. If you've ever tried to do your own data analysis, it's likely that you've used PCA, linear regression or LASSO. Hopefully, there were things about these algorithms that bothered you, but if not then, then I hope they do now because now you have the tools you need to deal with them!

In the course of these notes, worksheets, assignments, and discussions, we hope to have given you the tools to:

- Think about data **distributionally**
- **Estimate** interesting quantities using bootstrapping
- Generate **null hypotheses** to systematically test your results.

All of this can be done without (too much) theory and certainly without looking anything up in the back of a textbook. Instead, our framework requires a bit of coding know-how, a statistical and quantitative perspective, and a powerful (enough) computer.

Taken together, this course takes the perspective that a new kind of statistics and data-analysis needs to be taught, and practiced, in the era of modern computing. We hope we have given you an introduction to some of the most general purpose and powerful tools that you have access to currently. We sincerely hope you use these methods. Please reach out if there is some problem you are trying to solve and would like to bounce around some ideas.