Concepts and Computation: An Introduction to Political Methodology

Carlisle Rainey

2019-10-09

Contents

Chapter 1

Overview

Part I

Data Reduction

Chapter 2

Location and Scale

2.1 The Intuition

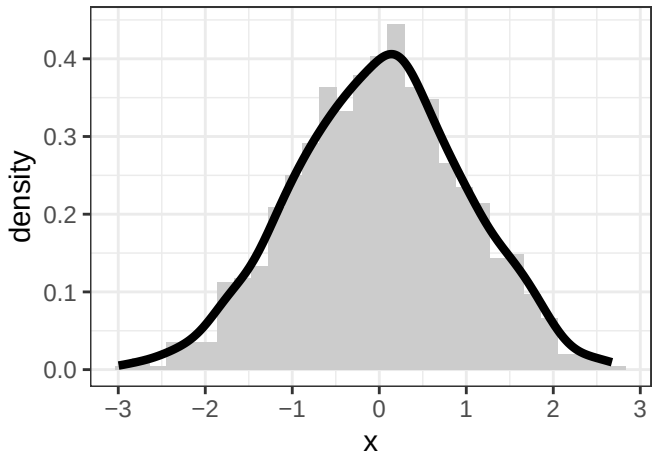
If we took a histogram and tried to describe it to someone else without showing it to them, the most **most** important pieces of information are usually the **location** and **scale**.[1]

We might describe the variable this way: "The values are about _____, give or take _____ or so." We can think of the first blank as the location and the second blank as the scale.

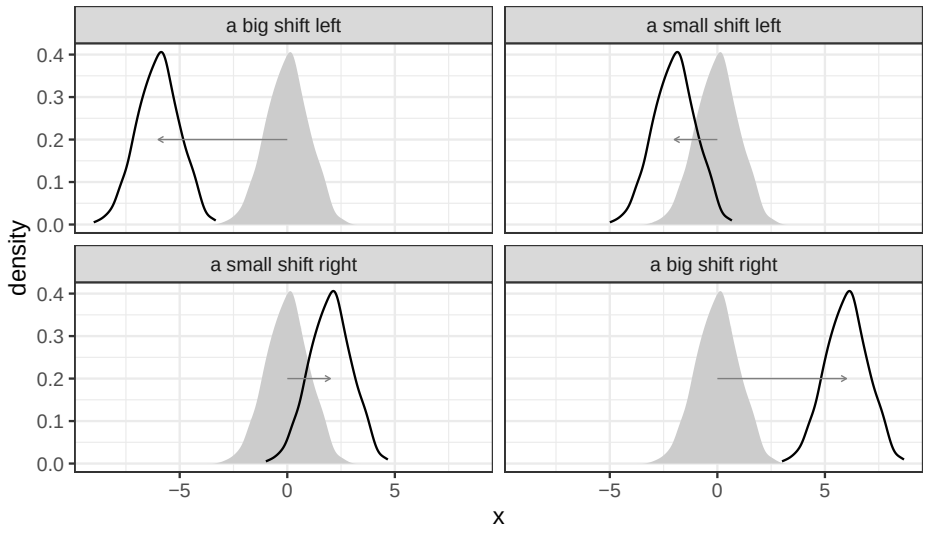
1. The **location** describes where the histogram is positioned along the left-right axis.
2. The **scale** describes the width (or "spread" or "dispersion") of the histogram.

Inspect the histogram of a hypothethical variable to the right. Notice the location and the scale. If we had to describe these data, we might say that our variable is "about zero give or take one or so."

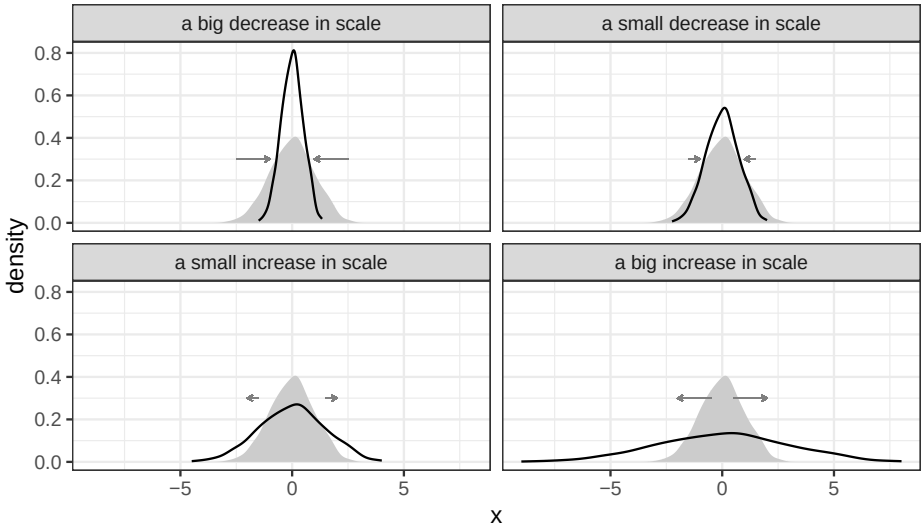
[1] I use these terms intentionally. Later, when we discuss random variables, the terms "location" and "scale" will return (with similar meanings). Indeed, we paramaterize many distributions according to their location and scale. For example, the normal distribution has a location parameter μ and a scale parameter σ.



While this variable has a particular location (about zero), we can imagine shifting it left or right. The figure below shows some possible shifts. We could shift it way to the left, so that it's "about -6" or a little bit to the right so that it's "about two."



We can also imagine increasing the scale (more spread) or decreasing the scale (less spread). The figure below shows some possible changes in scale. In each case, the "give or take" number is changing.



2.2 The Usual Measures

2.2.1 The Average

The most common measure of the location of a variable is the average.[2] Suppose we have a variable (a list of numbers) $X = \{x_1, x_2, ..., x_n\}$.

$$\text{average} = \frac{\text{the sum of the list}}{\text{the number of entries in the list}} = \frac{\sum_{i=1}^{n} x_i}{n}$$

The average is easy to compute and easy to work with mathematically.[3]

Unfortunately, the average doesn't have an easy interpretation. The best interpretation, in my mind, is as the balance-point for the data. If we imagine the left-right axis as a teeter-totter and stack the data along the beam according to their values, then the average is the position of the fulcrum that would balance the data-filled beam.

[2]Some people refer to the "average" as the "mean". I prefer to avoid this because the "mean" might also refer to the expectation of a random variable. I use "average" and "expected value" to differentiate these two meanings.

[3]The median, alternatively, is not easy to compute and quite difficult to work with mathematically.

2.2.2 The Standard Deviation

The most common measure of scale is the standard deviation (SD). The intuition is subtle, so let's look a a simple example. Rember, our goal is a "give-or-take number.

Suppose we have a list of numbers $X = \{1, 2, 3, 4, 5\}$. The average of this list is 3, so we can compute the *deviation from average* for each value.

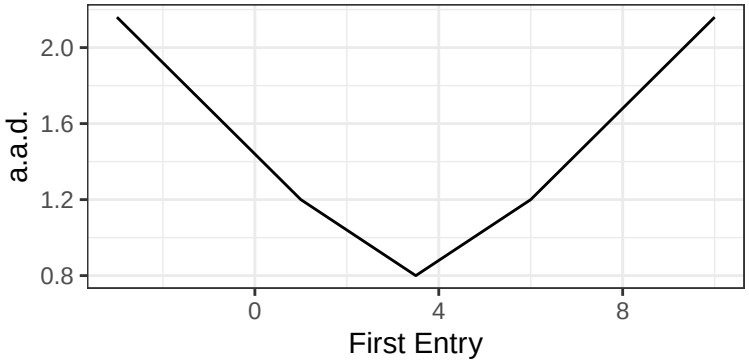
$$\text{deviation from average} = d = \text{value} - \text{average}$$

In this case, $d = \{-2, -1, 0, 1, 2\}$.

We want to use these deviations to find a give-or-take number.

Here's an initial idea. Just take the absolute values $|d| = \{2, 1, 0, 1, 2\}$. These tell us how far each entry falls away from the average. Then we could average the absolute devations to find how far a typical entries falls away from the average of the list. In this case, we get 1.2. This is reasonable approach and we'll refer to it as the average absolute devation or a.a.d. (It turns out that the a.a.d. isn't a common quantity, so I don't elevate it with an all-caps acronym.)

The a.a.d. has one big problem–it uses an absolute value. This introduces some computational and mathematical difficulties.[4]



So let's do something similar. Rather than take the absolute value, let's square the deviations, take the average, and then undo the square at the end, so that $\text{SD} = \sqrt{\text{avg}(d^2)}$.

Sometimes taking the (3) square root of (2) the average of (1) the squares is called the RMS. In this case, the **RMS of the deviations from the average is the SD**, so that

[4]Here's the gist: If you take an entry and slide it up and down (i.e., make it larger or smaller), then the a.a.d. moves up and down as well. This is fine, except the a.a.s. doesn't respond smoothly. The figure to the right shows what happens as we move the first entry on the list above around–notice the kink! The derivative of the a.a.d. isn't define here (i.e., there are lots of tangents). This makes things hard mathematically.

$$\text{SD} = \sqrt{\text{avg}(d^2)} = \sqrt{\frac{(x_i - \text{avg}(X))^2}{n}} = \text{RMS of deviations from average.}$$

The SD moves smoothly as you move around the entries in the list.

To calculate the SD, first make this little table, with the list of values, the deviations from the average, and the squares of the deviations.

X	d	d^2
1	-2	4
2	-1	1
3	0	0
4	1	1
5	2	4

Then compute the average of the squares of the deviations, which in this case is 2. Then take the square root of that average, which in this case is about 1.4. Notice that 1.4 is about 1.2 (the a.a.d.). The SD is bounded (weakly) below by the a.a.s., but they'll usually be close, so we can think of the SD as how far a typical point falls away from the average.

2.3 Robust Alternatives

The average and the SD are mathematically nice. But they are not robust. Seemingly innocuous changes in the variable can lead to large changes in the average and SD.[5]

We can definite robustness more concretely: How many observations do I need to corrupt to make the summary arbitrarily large?

Suppose the toy variable $X = \{0.1, -0.6, 1.1, 1.3, 0.2\}$. If I replace the first entry (0.1) with 1, 5, 10, 50, and so on, what happens to the average and SD? The table below shows that we can easily manipulate the average and SD by changing only one data point. In this sense, the average and SD are **fragile**.

Summary	Average	SD
Actual Data Set	0.42	0.70
First entry of X replaced with 1	0.60	0.71
…with 5	1.40	1.92
…with 10	2.40	3.86

[5]The mathmatical ease and the substantive fragility are related.

Summary	Average	SD
…with 50	10.40	19.81
…with 100	20.40	39.81
…with 500	100.40	199.80
…with 1,000	200.40	399.80

If corrupted data present a problem, then what do we mean by "corrupt"? There are (at least) three ways to imagine corrupting a measurement.

1. First, perhaps we have a data entry error. While entering data in a spreadsheet, you entered the number 50,000 into the "hours spent watching the news per day" variable instead of the "income" variable.
2. Second, perhaps our measurement procedure is noisy. Suppose we are coding Twitter posts by their support or opposition to President Trump. Our algorithm might interpret a sarcastic take as support when it actually presented intense opposition.
3. Third, the substantive model might not apply to a particular observation. Take Clark and Golder's project as an eplxame. They suggest that SMD systems should only have two parties. Indeed, this is a strong theoretical equilibirum. However, it might take several elections to reach this equilibrium. Parties might take several years to coordinate and consolodate. If we include a new democracy in the data set, then we might consider these data "corrupted" since the conceptual model doesn't apply (yet).

The average and SD respond to even a small amount of corrupt data.

As an alternative to the average, we might use the median, which is more robust. The median is the/a number which splits the values in half, so that equal numbers of entries lie above and below the median.

We have two common robust alternatives to the SD. The interquartile range (IQR) is the difference between the 25th and 75th quantiles. The median absolute deviation (MAD) is the median of the absolute values of the deviations from the **median** (almost the a.a.d., but using the medians in place of averages). It turns out that multiplying the MAD by 1.4826 makes it similar to the SD in many dataset, so it's common to rescale it.

To illustrate the robustness of each of our measures of location and scale, let's imagine a variable with 10 observations $X = \{-1.1, 1.5, -1, -0.1, -1.1, 0, -0.4, 0, 0.8, 0.4\}$. Let's see how the measures change as we corrupt more and more of the observations.

Summary	% Corrupted	Average	SD	Median	IQR	MAD
Actual Data Set	0%	-0.10	0.81	-0.05	1.15	0.96
First entry of X replaced with 100	10%	10.01	30.01	0.00	1.03	0.89
First two entries…	20%	19.86	40.07	0.00	1.03	0.89

Summary	% Corrupted	Average	SD	Median	IQR	MAD
First three entries...	30%	29.96	45.85	0.20	75.28	0.89
First four entries...	40%	39.97	49.02	0.60	100.00	2.00
First five entries...	50%	50.08	49.92	50.40	99.90	73.54
First six entries...	60%	60.08	48.89	100.00	99.50	0.00

This table illustrates that while the average and SD respond to *any* corruption, the median, IQR, and MAD remain reasonable summaries of the uncorrupted variable with 40%, 20%, and 30% of the data corrupted, respectively. T

The percent of the data that one can corrupt before they can make the measure arbitrarily large is called the **breakdown point**. Here are the breakdown points for our measures:

Measure	Breakdown Point
Average	0%
SD	0%
Median	50%
IQR	25%
MAD	50%

As you can see, the median and the MAD are highly robust–they achieve the theoretical maximum breakdown point.

2.4 Computation in R

We can easily calculate all these measures of location and scale in R.[6]

```
# create variable x = {1, 2, 3, 4, 5}
x <- 1:5

# compute measures of location and scale
mean(x)  # average
```

```
## [1] 3
```

[6]For reasons I don't want to deal with now, R uses the formula $SD = \sqrt{\frac{(x_i - \text{avg}(X))^2}{n-1}}$ rather than $\sqrt{\frac{(x_i - \text{avg}(X))^2}{n}}$. This means that R's SD will be slightly larger than the SD with my formula. This difference will be tiny in data sets with a typical number of observations.

```
sd(x) # SD; see sidenote
```

```
## [1] 1.581139
```

```
median(x)  # median
```

```
## [1] 3
```

```
IQR(x)  # IQR
```

```
## [1] 2
```

```
mad(x)  # MAD, rescaled by 1.4826
```

```
## [1] 1.4826
```

```
mad(x, constant = 1)  # MAD, not rescaled
```

```
## [1] 1
```

The functions above work nicely for computing on whole variables. But in most cases, we are interested in comparing the summaries across groups.

Take the nominate data set for example.

```
# load packages
library(tidyverse)

# load nominate data
df <- read_rds("data/nominate.rds") %>%
  glimpse()
```

```
## Observations: 7,080
## Variables: 7
## $ congress <int> 100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100...
## $ chamber  <chr> "House", "House", "House", "House", "House", "House",...
## $ state    <chr> "AL", "AL", "AL", "AL", "AL", "AL", "AL", "AK", "AZ",...
## $ district <int> 2, 4, 3, 5, 6, 1, 7, 1, 2, 3, 5, 4, 1, 3, 1, 4, 2, 36...
## $ party    <fct> Republican, Democrat, Democrat, Democrat, Democrat, R...
## $ name     <chr> "DICKINSON, William Louis", "BEVILL, Tom", "NICHOLS, ...
## $ ideology <dbl> 0.398, -0.213, -0.042, -0.175, -0.060, 0.373, -0.085,...
```

For these data, we might want to know the average ideology for Republicans and Democrats. We could do it the hard way.

```
# create a data frame with only republicans
rep_df <- df %>%
  filter(party == "Republican")

# compute average
mean(rep_df$ideology, na.rm = TRUE)
```

```
## [1] 0.4213385
```

But this is tedious, especially if we wanted to do it by party and Congress.

To compute these summaries for lots of subsets of the data, we have the `group_by()/summarize()` workflow.

`group_by()` defines several groups in the data frame. The first argument is the data frame to group (but we'll `%>%` it in). The remaining arguments are the grouping variables. You can think if the groups as a footnote at the bottom of the data set that just mentions the variables that define the groups of interest. Whenever we act (in the wrangling sense) on the data set and the action makes sense in the context of groups, the action will happen by group.

After grouping, we use `summarize()` to create summaries for each group. The first argument is the data frame to summarize (but we'll `%>%` it in). The remaining arguments are the summarizes to compute. The names of the remaining arguments become variables in the resulting data frame.

```
smry_df <- df %>%
  # group by party and congress
  group_by(party, congress) %>%
  # compute all of our measures of location and scale
  summarize(average = mean(ideology, na.rm = TRUE),
            sd = sd(ideology, na.rm = TRUE),
            median = median(ideology, na.rm = TRUE),
            iqr = IQR(ideology, na.rm = TRUE),
            mad = mad(ideology, na.rm = TRUE),
            mad1 = mad(ideology, constant = 1, na.rm = TRUE)) %>%
  # quick look at our work
  glimpse()
```

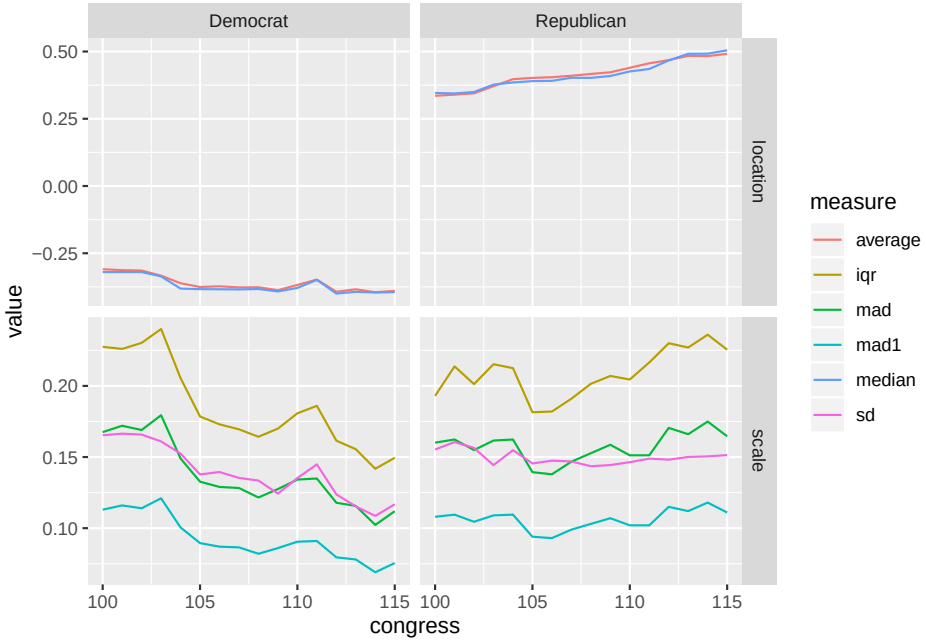
```
## Observations: 32
## Variables: 8
## Groups: party [2]
## $ party    <fct> Democrat, Democrat, Democrat, Democrat, Democrat, Dem...
## $ congress <int> 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110...
## $ average  <dbl> -0.3092901, -0.3130075, -0.3142407, -0.3333065, -0.36...
## $ sd       <dbl> 0.1653092, 0.1664293, 0.1658089, 0.1609726, 0.1524251...
## $ median   <dbl> -0.3200, -0.3200, -0.3200, -0.3360, -0.3815, -0.3835,...
## $ iqr      <dbl> 0.22750, 0.22600, 0.23025, 0.24000, 0.20550, 0.17850,...
## $ mad      <dbl> 0.1675338, 0.1719816, 0.1690164, 0.1793946, 0.1490013...
## $ mad1     <dbl> 0.1130, 0.1160, 0.1140, 0.1210, 0.1005, 0.0895, 0.087...
```

We can plot this measures to get a sense of how they change over time. Notice that `mad` (rescaled by multiplying by 1.4826) closely corresponds to the SD, but `mad1` (not rescaled) is much smaller.

```
# wrangle the data for plotting
gg_df <- smry_df %>%
  pivot_longer(average:mad1, names_to = "measure") %>%
  mutate(measure_of = ifelse(measure %in% c("average", "median"), "location", "scale"))
  glimpse()
```

```
## Observations: 192
## Variables: 5
## Groups: party [2]
## $ party      <fct> Democrat, Democrat, Democrat, Democrat, Democrat, D...
## $ congress   <int> 100, 100, 100, 100, 100, 100, 101, 101, 101, 101, 1...
## $ measure    <chr> "average", "sd", "median", "iqr", "mad", "mad1", "a...
## $ value      <dbl> -0.3092901, 0.1653092, -0.3200000, 0.2275000, 0.167...
## $ measure_of <chr> "location", "scale", "location", "scale", "scale", ...
```

```
# plot the measures of location and scale
ggplot(gg_df, aes(x = congress, y = value, color = measure)) +
  geom_line() +
  facet_grid(cols = vars(party), rows = vars(measure_of), scales = "free_y")
```



Chapter 3

The Normal Model

3.1 The Intuition

Last week, we used the average and SD to reduce and entire variable to two summaries. We use the average and SD to fill in the following sentence: "The values are about _____, give or take _____ or so."

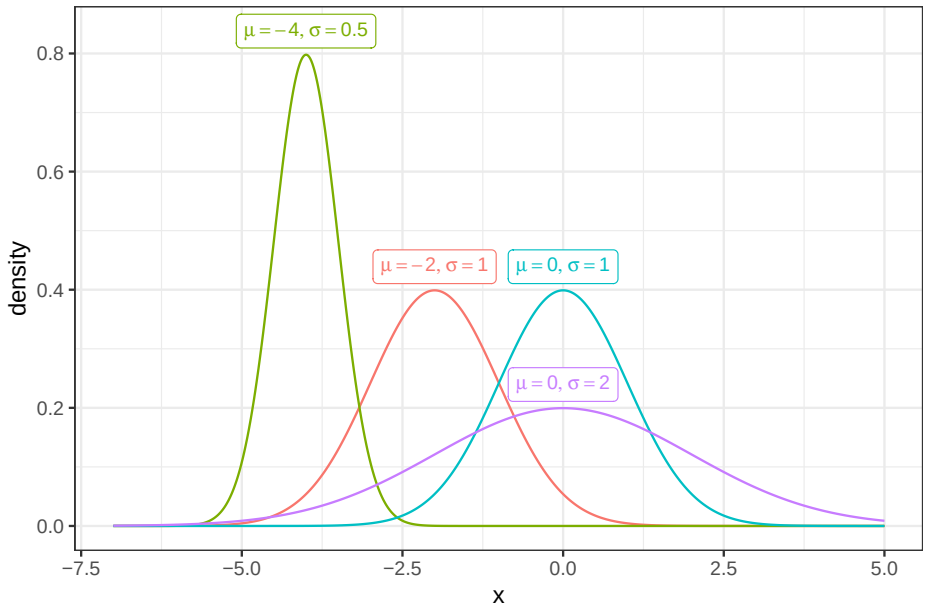
This week, we add an additional assumption. This week, we also say that the histogram of the variable follows the normal curve. The normal curve is a bell-shaped curve with a particular equation. There are two varieties. There is a general, parameterized normal distibution that can move left and right (i.e., change location) and grow wider or taller (i.e., change scale)

3.2 The Normal Curve(s)

There are two particular normal curves that we care about

1. **the normal curve**, which has a location and scale parameter that we can specify: $f(x|\mu,\sigma) = \phi(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$
2. **the *standard* normal curve**, with the location and scale parameters fixed: $f(x|\mu=0,\sigma=1) = \phi(x|\mu=0,\sigma=1) = \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}}$

These equations are complicated. Instead of memorizing them or working carefully through the math, just understand (for now) that the normal curve has an equation that exactly characterizes it. The figure below shows the *standard* normal curve ($\mu = 0$ and $\sigma = 1$) and several other paramaterizations.

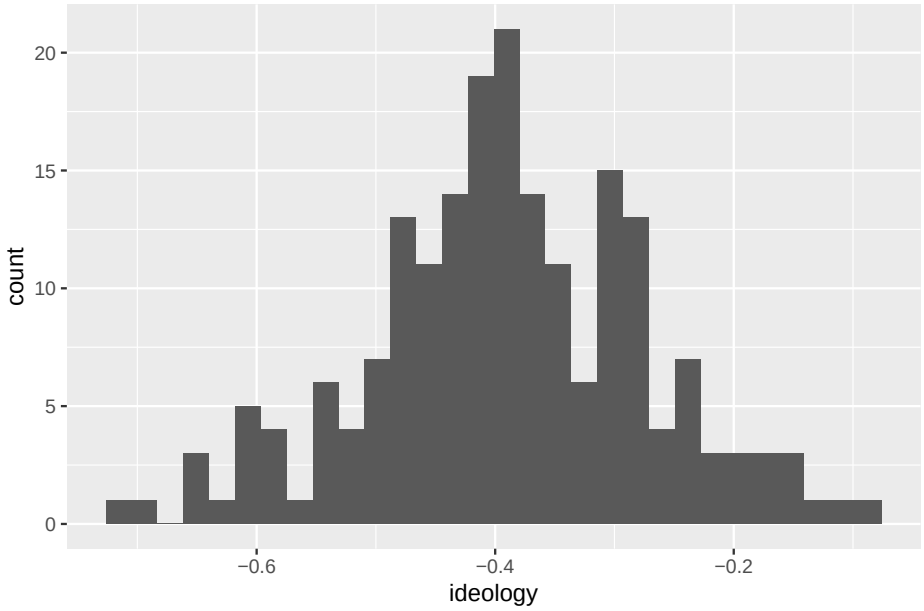


3.3 The Empirical Rule

It turns out that many variable's have a histogram that resembles the normal curve. Because of this, the normal curve can sometimes serve as an effective model for these variables.

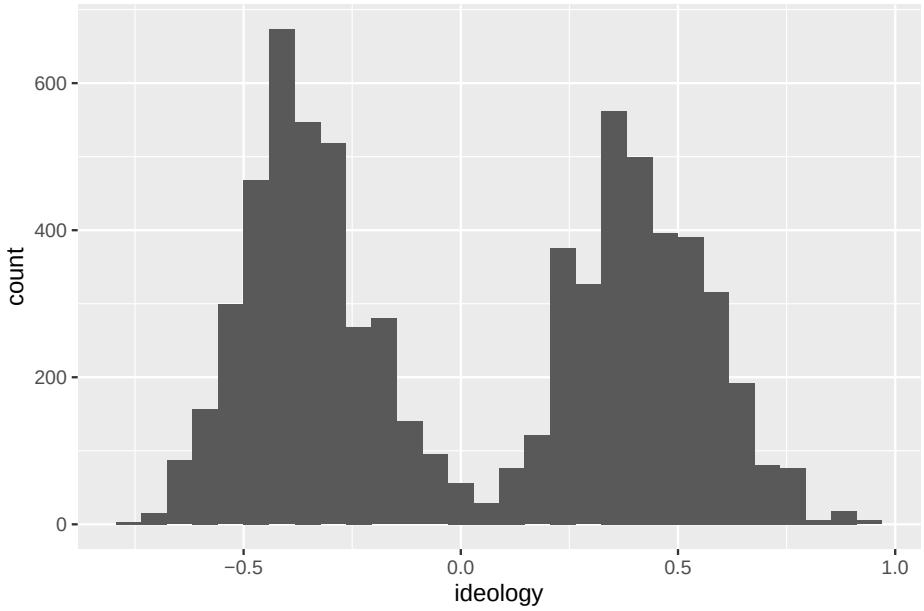
For example, NOMINATE ideology scores for Republicans in the 115th Congress roughly follow the normal curve.

```
df <- read_rds("data/nominate.rds") %>%
  filter(party == "Democrat", congress == 115)

ggplot(df, aes(x = ideology)) +
  geom_histogram()
```



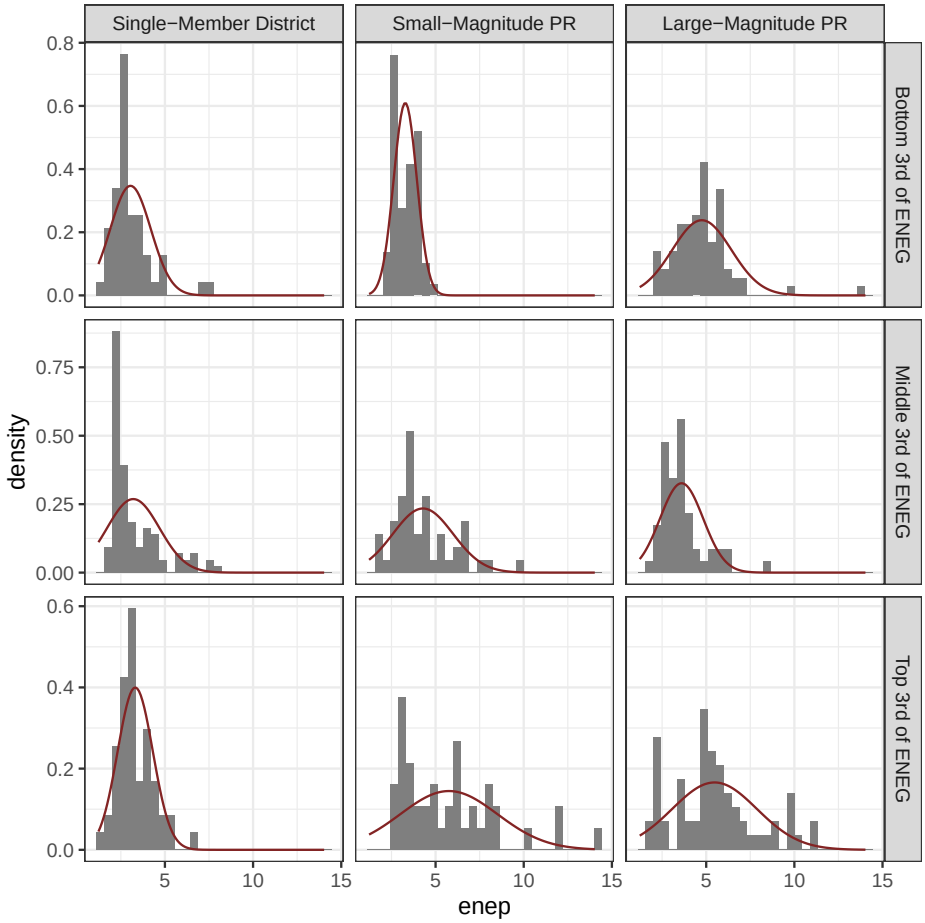
However, the ideology scores for both Republicans and Democrats together does not follow a normal curve.

```
df <- read_rds("data/nominate.rds")

ggplot(df, aes(x = ideology)) +
  geom_histogram()
```



The histograms of ENEP by electoral system and social heterogeneity deviate slightly from the normal curve.

```
## Observations: 1,161
## Variables: 4
## $ x                   <dbl> 1.23, 1.33, 1.43, 1.53, 1.63, 1.73, 1.83,...
## $ density             <dbl> 0.02616495, 0.02960818, 0.03338541, 0.037...
## $ electoral_system    <fct> Large-Magnitude PR, Large-Magnitude PR, L...
## $ social_heterogeneity <fct> Bottom 3rd of ENEG, Bottom 3rd of ENEG, B...
```



If the variable seems to follow the normal curve, then we have the following rules:

- About *68%* of the data (i.e., "most") fall within **1 SD** of the average.
- About *95%* of the data (i.e., "almost all") fall within **2 SDs** of the average.

We can evaluate this rule with the parties data above. Some of the nine histograms follow the normal curve quite well (e.g., lower-left). Other's seem to

meaningfully deviate from the normal curve (e.g., middle-left).

The table below shows the actual percent of the variable that falls within one and two SDs of the average for each histogram. As you can see, for the lower-left panel (SMD, Top 3rd), the empircal rule of 68% and 95% matches the actual values of 74% and 98% fairly well. For the middle-left panel (SMD, Middle 3rd), the empirical rule matches the actual values of 87% and 93% less well.

Across all histograms, it seems fair that the empirical rule works as a rough approximation, even for histograms that meaningfully deviate from the normal curve.

Electoral System	Social Heterogeneity	within 1 SD	within 2 SDs
Single-Member District	Bottom 3rd of ENEG	87%	96%
Single-Member District	Middle 3rd of ENEG	87%	93%
Single-Member District	Top 3rd of ENEG	74%	98%
Small-Magnitude PR	Bottom 3rd of ENEG	68%	97%
Small-Magnitude PR	Middle 3rd of ENEG	73%	96%
Small-Magnitude PR	Top 3rd of ENEG	76%	93%
Large-Magnitude PR	Bottom 3rd of ENEG	80%	98%
Large-Magnitude PR	Middle 3rd of ENEG	77%	96%
Large-Magnitude PR	Top 3rd of ENEG	65%	97%

3.4 The Normal Approximation

If our normal model summarizes a histogram well, then we can use the model to estimate the percent of the observations that fall in a given range. There are two approaches:

Just like we add up the area of the bars to compute percentages with a histogram, **we add up the area under the normal curve to approximate percentages.**

1. Use a normal table from a textbook. Because the table is for the standard normal curve, we need to **re-locate and re-scale the data to fit the standard normal curve**.
2. Use the `pnorm()` function in R. Because this function is parameterized with location and scale, we can simple **re-locate and re-scale the curve to fit the data**.

3.4.1 Normal Table

Normal tables offer an antiquated method to use the normal distribution to approximate percentages. Because we cannot have a normal table for all possible

locations and scales, we have one: the standard normal table, which works for a variable with an average of zero and an SD of one.

This seems limiting, but it turns out that we can easily re-locate and re-scale any value to match the standard normal curve. We simply subtract the average and divide by the SD. We call this new value a z-score.

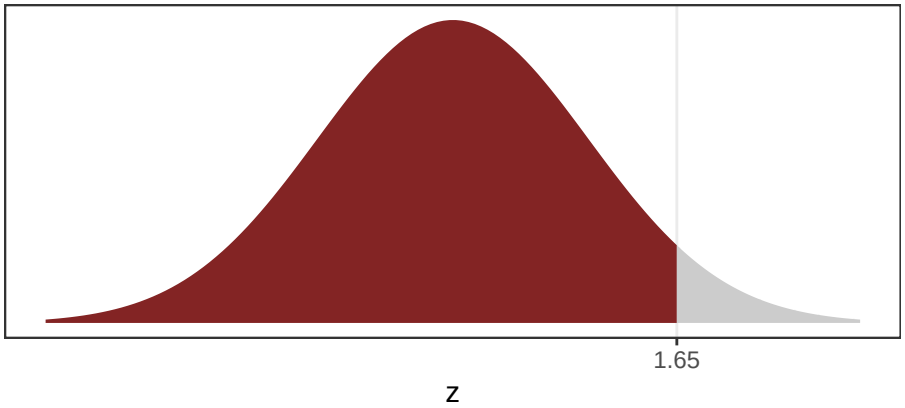
$$z\text{-score} = \frac{\text{value} - \text{average}}{\text{SD}}$$

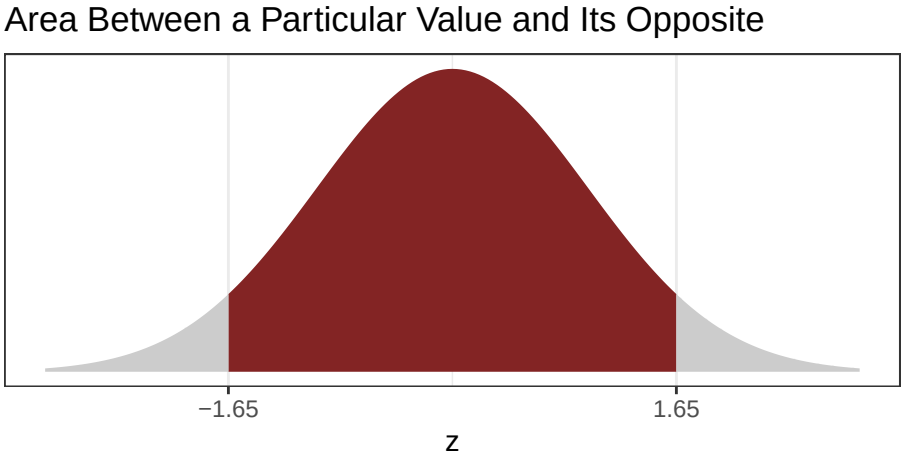
FSuppose we have the list $X = \{1, 2, 3, 4, 5\}$. Then the average is 3, and the SD is about 1.26. We can compute the $z$score for the first entry 1 as $\frac{1-3}{1.25} \approx -1.58$. Similarly, we can convert the entire list to z-scores and get $Z = \{1.59, -0.79, 0.00, 0.79, 1.59\}$. If you compute the average and SD of the list Z, you will find zero and one, respectively.

We can then use a normal table to compute areas under the normal curve between (or above or below) these values of z. There are two types of normal tables.

1. Some tables report the percent (or proportion) of the normal curve **below** a particular value z.
2. Other tables report the percent (or proportion) of the normal curve **between** a particular value z and $-z$. (The normal table on p. A-104 of FPP works this way.)

Area Below a Particular Value





Either table works, but you must know what type of table you are working with. Depending on the question, one type might offer a more direct solution.

Here's a small normal table for a few values of z that uses both approaches.

z	% less than z	% between -z and z	Status
0.00	50%	0%	
0.10	54%	8%	
0.20	58%	16%	
0.30	62%	24%	
0.40	66%	31%	
0.50	69%	38%	
0.75	77%	55%	
1.00	84%	68%	Important
1.50	93%	87%	
1.64	95%	90%	Important
1.96	98%	95%	Important
2.00	98%	95%	Important
3.00	100%	100%	

In order to use the table to find the area between any two values, you need to use the following three rules in combination.

1. The normal table gives the area (i) below z or (ii) between $-z$ and z.
2. The area under the entire normal curve is 1 or 100%.
3. The normal curve is symetric, so that the area to the right of z equals the area to the left of $-z$.

3.4.2 pnorm()

The `pnorm()` function in R return the area under the normal curve less than z. By default, it uses the standard normal curve, but you can specify a `mean` and `sd` if you prefer to re-locate and/or re-scale the curve to fit your values.

```
# area under the std. normal curve less than 1
pnorm(1)
```

```
## [1] 0.8413447
```

```
# area under the a normal curve (with average of 1 and SD of 4) less than 1
pnorm(1, mean = 1, sd = 4)
```

```
## [1] 0.5
```

```
# area between -1.64 and 1.64
pnorm(1.64) - pnorm(-1.64)
```

```
## [1] 0.8989948
```

3.4.3 Exactly Percentages

To actually compute percentages, we can create a function that works just like `pnorm()`, but it returns the percent *of the data* that fall below a particular value. The most convenient method is to create an "empirical cumulative distribution function*.

This function is somewhat confusing. The `ecdf()` function does not return the proportion below its argument. Instead, it creates a function that returns the percent below its argument. If we have a numeric vector `x`, then `ecdf(x)` is a function! Let that settle in… both `ecdf` and `ecdf(x)` are function. The function `ecdf` (I'm dropping the `()` for clarity) is a function *that creates a function*, and `ecdf(x)()` (I'm including the `()`, as usual, for clarity) is a function that returns the percent below.

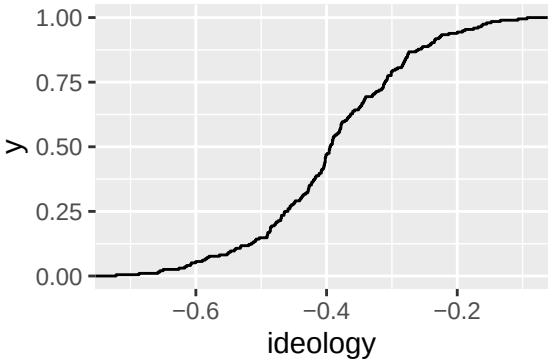
```
df <- read_rds("data/nominate.rds") %>%
  filter(party == "Democrat", congress == 115)

# normal approximation for % of Democrats less than -0.05
avg <- mean(df$ideology)
sd <- sd(df$ideology)
pnorm(-0.5, mean = avg, sd = sd)
```

```
## [1] 0.1731597
```

```
# exact % of Democrats less than -0.05
ecdf(df$ideology)(-0.5)
```

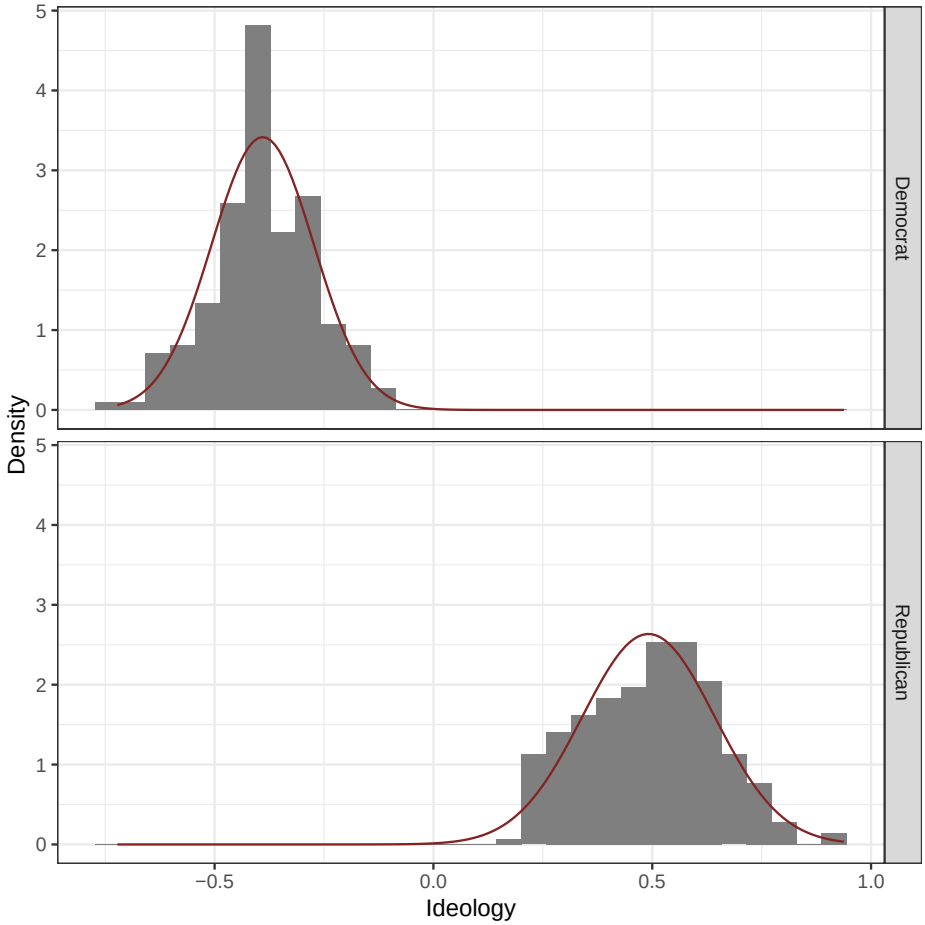
```
## [1] 0.1479592
```

We can also plot the ECDF with ggplot2.



3.5 Review Exercises

The plot below show the histograms for the ideology of legislators in the U.S. House by party.



We can compute the average and SD by party.

Party	Average	SD
Democrat	-0.39	0.12
Republican	0.49	0.15

The table below lists some of the leaders of each party and their ideology score. For each leader, use our three approaches to compute the percent of the party that is "more extreme" than their leader: inspect the histogram, use the normal approximation, and use R to compute the answer exactly.

Name	Party	Position	Ideology Score	Inspect Histogram	Normal Approximation	Actual
RYAN, Paul D.	Republican	Speaker of the House	0.56			

Name	Party	Position	Ideology Score	Inspect Histogram	Normal Approximation	Actual
MCCARTHY, Kevin	Republican	Majority Leader	0.46			
SCALISE, Steve	Republican	Majority Whip	0.56			
McMORRIS RODGERS, Cathy	Republican	Conference Chair	0.43			
PELOSI, Nancy	Democrat	Minority Leader	-0.49			
HOYER, Steny Hamilton	Democrat	Minority Whip	-0.38			
CLYBURN, James Enos	Democrat	Assistant Democratic Leader	-0.46			
LEWIS, John R.	Democrat	Senior Chief Deputy Minority Whip	-0.59			