

Introduction to Statistical Inference with R

An R Companion to

Introduction to Statistical Investigations (Preliminary Edition)

Randall Pruim and Lana Park

May 19, 2015

Contents

0 Preliminaries	5
0.0 Getting Started with R and RStudio	5
0.1 Introduction to the Six-Step Method	7
0.2 Exploring Data	11
0.3 Exploring random Processes	15
0.4 Other Visualizations	17
1 Significance: How strong is the evidence?	19
1.1 Introduction to Chance Models	19
1.2 Measuring the Strength of Evidence	22
1.3 Alternative Measure of Strength of Evidence	25
1.4 What Impacts Strength of Evidence?	30
1.5 Inference on a single proportion: Theory-based approach	37
2 Generalization: How Broadly Do the Results Apply?	49
2.1 Sampling from a Finite Population	49
2.2 Inference for a Single Quantitative Variable	57
2.3 Errors and Significance	69
3 Estimation: How Large is the Effect?	73
3.1 Statistical Inference - Confidence Intervals	73
3.2 2SD and Theory-Based Confidence Intervals for a Single Proportion	81

3.3	2SD and Theory-Based Confidence Intervals for a Single Mean	89
3.4	Factors That Affect the Width of a Confidence Interval	92
3.5	Cautions When Conducting Inference	98
4	Causation: Can We Say What Caused the Effect?	107
4.1	Association and Confounding	107
4.2	Observational studies versus experiments	108
5	Comparing Two Proportions	109
5.1	Comparing Two Groups: Categorical Response	109
5.2	Comparing Two Properties: Simulation-Based Approach	113
5.3	Comparing Two Proportions: Theory-Based Approach	217
6	Comparing Two Means	257
6.1	Comparing Two Groups: Quantitative Response	257
6.2	Comparing Two Means: Simulation-Based Approach	260
6.3	Comparing Two Means: Theory-Based Approach	266
7	Paired Data: One Quantitative Variable	273
7.1	Paired Designs	273
7.2	Simulation-Based Approach for Analyzing Paired Data	273
7.3	Theory-Based Approach to Analyzing Data from Paired Samples	281
8	Comparing More Than Two Proportions	287
8.1	Simulation-Based Approach to Compare Multiple Proportions	287
8.2	Theory-Based Approach to Compare Multiple Proportions	291
9	Comparing More than Two Means	303
9.1	Simulation-Based Approach for Comparing More than Two Groups with a Quantitative Response	303
9.2	Theory-based Approach to Comparing More than Two Groups with a Quantitative Response . .	306
10	Two Quantitative Variables	313
10.1	Summarizing the Relationship Between Two Quantitative Variables Using the Correlation Coefficient	313
10.2	Inference for the Correlation Coefficient: A Simulation-based Approach	316
10.3	Least Squares Regression	321

10.4 Inference for Regression Slope: Simulation-Based Approach	326
10.5 Inference for the Regression Slope: Theory-Based Approach	329

0.0 Getting Started with R and RStudio

R is divided up into packages. A few of these are loaded every time you run R, but most have to be selected. This way you only have as much of R as you need.

In the Packages tab, check the boxes next to the following packages to load them:

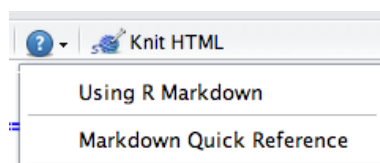
- **mosaic** (a package from Project MOSAIC)
- **ISlwithR** (data sets)

RStudio provides several ways to create documents that include text, R code, R output, graphics, even mathematical notation all in one document. The simplest of these is R Markdown.

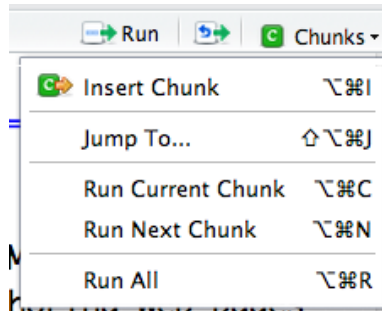
To create a new R Markdown document, go to “File”, “New”, then “R Markdown.”

When you do this, a file editing pane will open with a template inserted. If you click on “Knit HTML”, RStudio will turn this into an HTML file and display it for you. Give it a try. You will be asked to name your file if you haven’t already done so. If you are using the RStudio server in a browser, then your file will live on the server (“in the cloud”) rather than on your computer.

If you look at the template file you will see that the file has two kinds of sections. Some of this file is just normal text (with some extra symbols to make things bold, add in headings, etc.) You can get a list of all of these mark up options by selecting the “Markdown Quick Reference” in the question mark menu.



The second type of section is an R code chunk. These are colored differently to make them easier to see. You can insert a new code chunk by selecting “Insert Chunk” from the “Chunks” menu:



(You can also type ```{r}` to begin and ```` to end the code chunk if you would rather type.) You can put any R code in these code chunks and the results (text output or graphics) as well as the R code will be displayed in your HTML file.

There are options to do things like (a) run R code without displaying it, (b) run R code without displaying the output, (c) controlling size of plots, etc., etc. But for starting out, this is really all you need to know.

R Markdown files are self-contained, meaning they do not have access to things you have done in your console. (This is good, else your document would change based on things not in the file.) This means that you must explicitly load data, and require packages *in the R Markdown file* in order to use them. For this text, this means that most of your R Markdown files will have a chunk near the beginning that includes

```
require(mosaic) # load the mosaic package
```

Functions in R use the following syntax:

```
functionname(argument1, argument2, ...)
```

function-syntax

The arguments are always surrounded by (round) parentheses and separated by commas.

Some functions (like `data()`) have no required arguments, but you still need the parentheses.

Most of what we will do in the subsequent chapters makes use of a single R template:

```
[ ] ( [ ] ~ [ ] , data = [ ] )
```

It is useful if we name the slots in this template:

```
goal ( [y] ~ [x] , data = [mydata] )
```

However, there are some variations on this template:

```
### Simpler version
goal(~x, data = mydata)
### Fancier version:
goal(y ~ x | z, data = mydata)
### Unified version:
goal(formula, data = mydata)
```

To use the template, you just need to know what goes in each slot. This can be determined by asking yourself two questions:

1. What do you want R to do?

- this determines what function to use (goal).

2. What must R know to do that?

- this determines the inputs to the function
- for describing data, must must identify *which data frame* and *which variable(s)*.

Further, if you begin a command and hit the TAB key, R will show you a list of possible ways to complete the command. If you hit TAB after the opening parenthesis of a function, it will show you the list of arguments it expects. The up and down arrows can be used to retrieve past commands.

Additional R functionality will be introduced as we go along. The **mosaic** package includes several vignettes with additional information about using the package and using R.

0.1 Introduction to the Six-Step Method

Example P.1: Organ Donations

Now that we've explained a few basics for using R, let's take a look at a data set.

Data sets in R are usually stored as **data frames** in a rectangular arrangement with rows corresponding to **observational units** and columns corresponding to **variables**. A number of data sets are built into R and its packages. The package for our text is **ISIwithR** which comes with a number of data sets.

```
require(ISIwithR) # tell R to use the package for our textbook
data(OrganDonor) # load the OrganDonor dataset
```

If you want a list of all data sets available to you in loaded packages, use **data()** without any arguments. If you want to view the entire data set, just typing the name will show the details in the console.

```
data() # list all datasets available in loaded packages
OrganDonor # show entire dataset in console
```

For large data sets, it may be more practical to look at different types of summaries or subsets of the data.

```
head(OrganDonor) # first six cases of the dataset
```

```
default choice
1 opt-in donor
2 opt-in donor
3 opt-in donor
4 opt-in donor
5 opt-in donor
6 opt-in donor
```

```
summary(OrganDonor) # summary of each variable
```

```
default      choice
Length:161    Length:161
Class :character Class :character
Mode :character Mode :character
```

```
str(OrganDonor)           # structure of the dataset

'data.frame': 161 obs. of  2 variables:
 $ default: chr  "opt-in" "opt-in" "opt-in" "opt-in" ...
 $ choice : chr  "donor"  "donor"  "donor"  "donor"  ...

dim(OrganDonor)           # number of rows and columns

[1] 161  2

nrow(OrganDonor)          # number of rows

[1] 161

ncol(OrganDonor)          # number of columns

[1] 2
```

Now that we have a general sense of how the data is structured, we can take a more detailed look by using the R template. Let's say we want a count of observational units of each variable. We can tally the number by using the `tally()` function.

```
tally(~choice, data = OrganDonor)

donor  not
  108   53

tally(~default, data = OrganDonor)

neutral opt-in opt-out
   56    55    50
```

This didn't really show us any more information than the `summary()` from above so instead, let's tally the variables together.

```
tally(~choice + default, data = OrganDonor)

      default
choice neutral opt-in opt-out
donor    44    23    41
not      12    32     9

tally(~choice + default, data = OrganDonor, margins = TRUE)

      default
choice neutral opt-in opt-out Total
donor    44    23    41    108
not      12    32     9     53
Total    56    55    50    161
```

Notice that the default for `tally()` was to exclude the total counts of each row and column. You could have used either tab completion or search `tally()` in the help section to find `margins` and set `margins=TRUE`. There will be many instances where you will need to change the default settings of a function.

Moreover, we can change the organization of the variables for a slightly different output:

```
tally(choice ~ default, data = OrganDonor)
```

	default		
choice	neutral	opt-in	opt-out
donor	44	23	41
not	12	32	9


```
tally(choice ~ default, data = OrganDonor, format = "percent")
```

	default		
choice	neutral	opt-in	opt-out
donor	78.57143	41.81818	82.00000
not	21.42857	58.18182	18.00000

This may be a little confusing now (proportions will be covered in chapter 2) but let's focus more on the the changed organization of the variables in the `tally()` function. This version of tallying calculated the proportions (and percentages) of participants who agreed and did not agree to become organ donors (`choice`) in each of the groups opt-in, opt-out, and neutral (`default`).

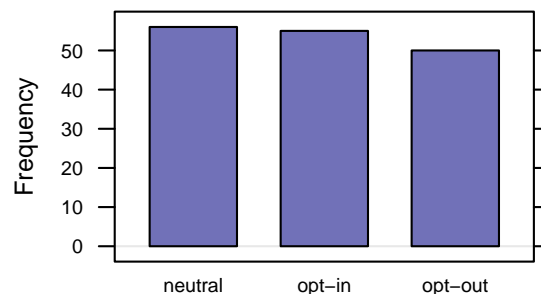
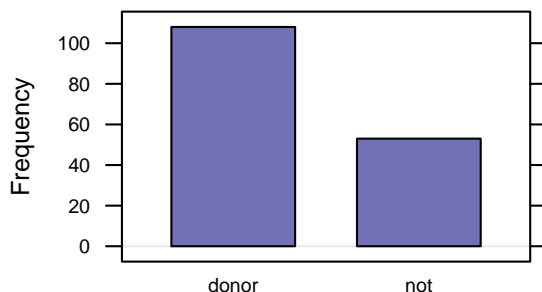
R also has many tools to visualize data. The general syntax for making a graph of one variable in a data frame is

```
plotname(~variable, data = dataName)
```

In other words, there are three pieces of information we must provide to R in order to get the plot we want:

- The kind of plot (`histogram()`, `bargraph()`, `densityplot()`, `bwplot()`, etc.)
- The name of the variable
- The name of the data frame this variable is a part of.

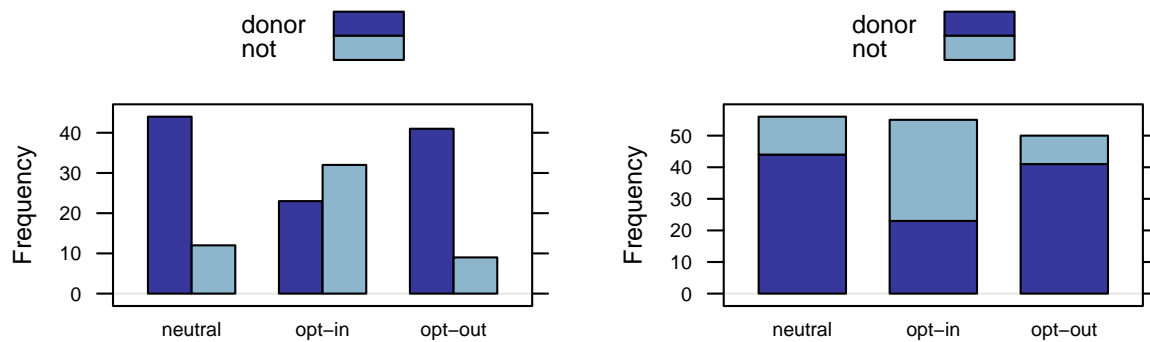
```
bargraph(~choice, data = OrganDonor)
bargraph(~default, data = OrganDonor)
```



Notice that the `bargraph()` uses the frequency, or counts.

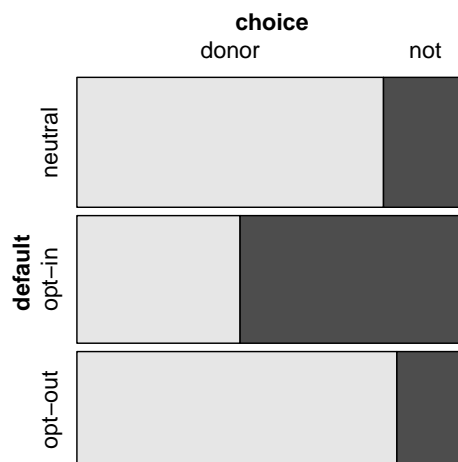
In order to graph the variable `default` and show what `choice` each option made, we can utilize the argument `groups=`.

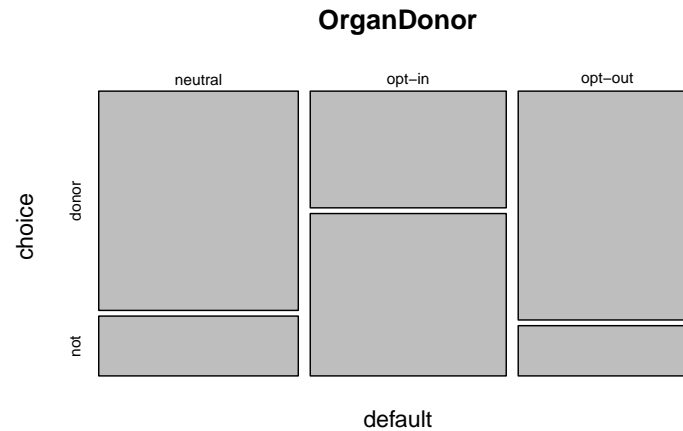
```
bargraph(~default, groups = choice, data = OrganDonor, auto.key = TRUE)
bargraph(~default, groups = choice, data = OrganDonor, auto.key = TRUE, stack = TRUE)
```



Although the bargraph is useful, the y-axis shows counts and not the percentages as in the text. The function `mosaic()` or `mosaicplot()` plots the variables relative to each other, in a way that reveals proportions, or percentages.

```
mosaic(choice ~ default, data = OrganDonor)
mosaicplot(default ~ choice, data = OrganDonor)
```





0.2 Exploring Data

Example P.2: Old Faithful

Everytime you use a new data set, it is beneficial to look at a some key summary statistics.

```
head(OldFaithful1)
```

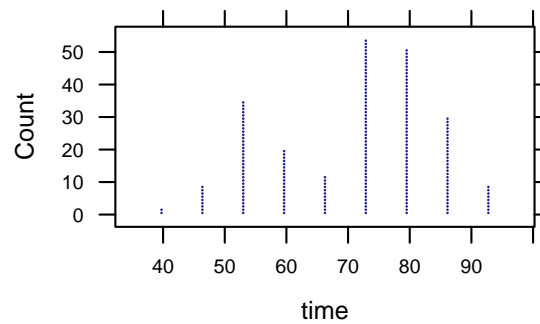
```
  time
1   55
2   58
3   56
4   50
5   51
6   60
```

```
summary(OldFaithful1)
```

```
  time
Min.  :42.00
1st Qu.:60.00
Median :75.00
Mean   :71.01
3rd Qu.:81.00
Max.   :95.00
```

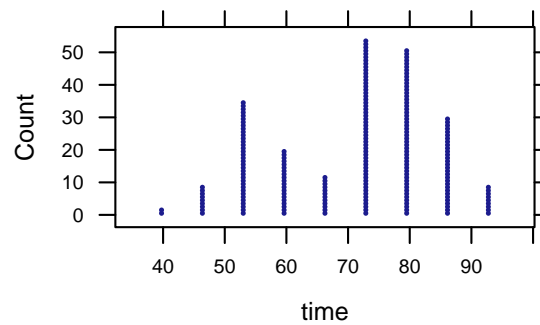
Another useful graph for examining the **shape**, **center**, and **variability** is the **dotplot**:

```
dotPlot(~time, data = OldFaithful1)
```



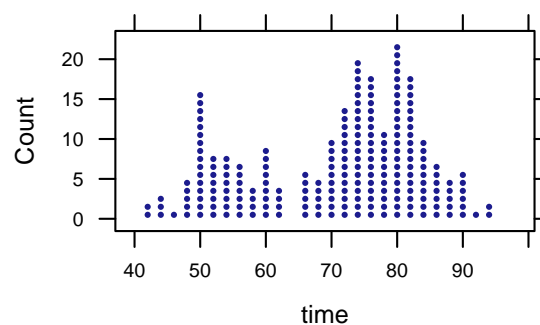
The dots in this plot are a bit small. The defaults for `dotPlot()` may not be the best way to examine a particular data set. We can increase the size of the dots using the `cex` argument. (`cex` stands for “character expansion” and is used to scale up or down the size of plotting characters – in this case the dots.)

```
dotPlot(~time, data = OldFaithful1, cex = 2)
```



Or we can change the distance between columns of dots

```
dotPlot(~time, data = OldFaithful1, width = 2)
```

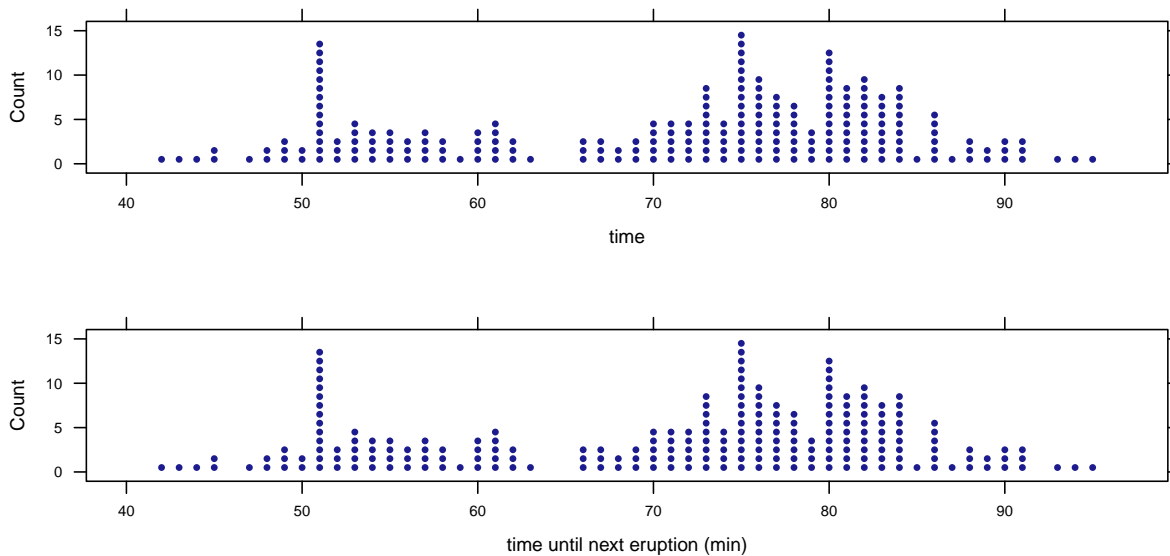


Notice that the dots have been automatically resized when we do this.

The appropriate choice may depend on the intended size and shape of the plot. The plots below are much wider, allowing us to present a finer view of the data. In the second plot, we have also added a more informative label.

```
dotPlot(~ time, data = OldFaithful1, width = 1)
dotPlot(~ time, data = OldFaithful1, width = 1,
        xlab = "time until next eruption (min)")
```

Figure P.3



Similar to the bargraph, we can organize the variables a little differently for the dotplot to graph them in relation to one another.

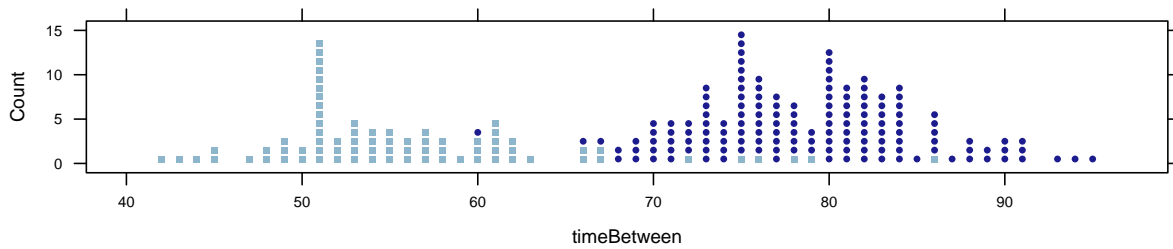
```
head(OldFaithful2)
```

	eruptionType	timeBetween
1	short	55
2	short	58
3	short	56
4	short	50
5	short	51
6	short	60

```
summary(OldFaithful2)
```

eruptionType	timeBetween
Length:222	Min. :42.00
Class :character	1st Qu.:60.00
Mode :character	Median :75.00
	Mean :71.01
	3rd Qu.:81.00
	Max. :95.00

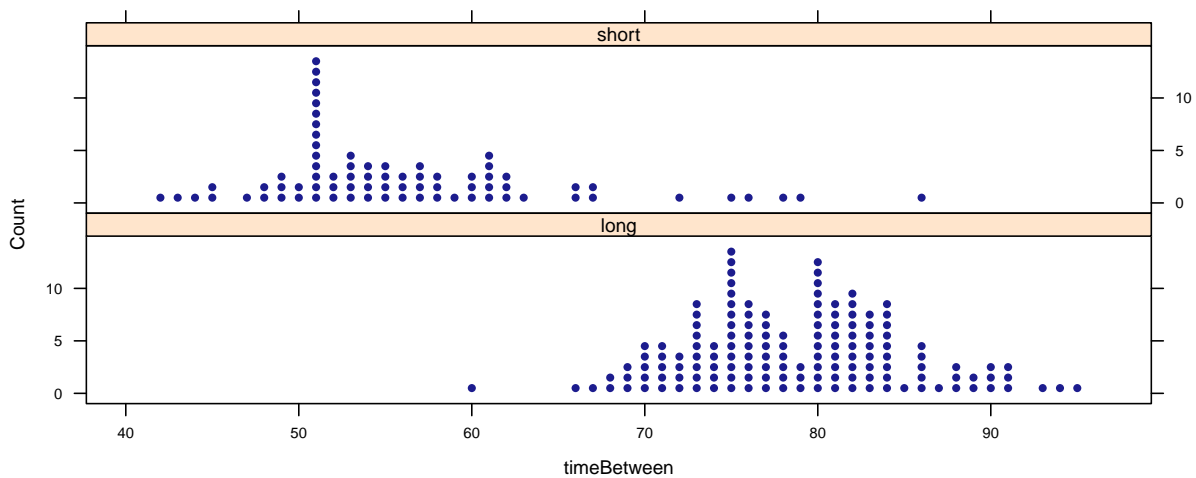
```
dotPlot(~timeBetween, groups = eruptionType, data = OldFaithful2, width = 1)
```



The formula for a **lattice** plot can be extended to create multiple panels (sometimes called **facets**) based on a “condition”, often given by another variable. This is another way to look at multiple groups simultaneously. The general syntax for this becomes

```
plotname(~variable | condition, data = dataName)
```

```
dotPlot(~timeBetween | eruptionType, data = OldFaithful2, width = 1, layout = c(1, 2))
```



For more key numerical summaries of the data set, we can use the **favstats()** for “favorite” statistics.

```
favstats(~timeBetween, data = OldFaithful2)
```

	min	Q1	median	Q3	max	mean	sd	n	missing
	42	60	75	81	95	71.00901	12.79918	222	0

```
favstats(timeBetween ~ eruptionType, data = OldFaithful2)
```

	eruptionType	min	Q1	median	Q3	max	mean	sd	n	missing
1	long	60	75	78.5	83.00	95	78.69178	6.251692	146	0
2	short	42	51	54.0	60.25	86	56.25000	8.457147	76	0

TableP1

Here are ways to find the mean and the standard deviation separately:

```
mean(~timeBetween, data = OldFaithful2)
```

```
[1] 71.00901
```

```
sd(~timeBetween, data = OldFaithful2)
```

```
[1] 12.79918
```

```
mean(timeBetween ~ eruptionType, data = OldFaithful2)
```

```
      long      short  
78.69178 56.25000
```

```
sd(timeBetween ~ eruptionType, data = OldFaithful2)
```

```
      long      short  
6.251692 8.457147
```

```
mean(~timeBetween | eruptionType, data = OldFaithful2)
```

```
      long      short  
78.69178 56.25000
```

```
sd(~timeBetween | eruptionType, data = OldFaithful2)
```

```
      long      short  
6.251692 8.457147
```

0.3 Exploring random Processes

Exploration P.3: Cars or Goats

The **mosaic** package has a function **rflip()** that **simulates** coin tosses. We define arguments **n** (the number of flips) and **prob** (the probability of heads).

```
rflip(n = 1, prob = 0.5)
```

```
Flipping 1 coin [ Prob(Heads) = 0.5 ] ...
```

```
T
```

```
Number of Heads: 0 [Proportion Heads: 0]
```

```
rflip(n = 5, prob = 0.5)
```

```
Flipping 5 coins [ Prob(Heads) = 0.5 ] ...
H T H H H
Number of Heads: 4 [Proportion Heads: 0.8]
```

Although `rflip()` simulates coin tosses, where the probability of heads should be 0.5, we can also simulate any **random process** by changing the **probability**.

```
rflip(n = 15, prob = 1/3)

Flipping 15 coins [ Prob(Heads) = 0.33333333333333 ] ...
T T T H T H T H T H T T T
Number of Heads: 5 [Proportion Heads: 0.33333333333333]
```

This is equivalent to the playing 15 games (flips), each game having a 1/3 chance of picking the car (heads).

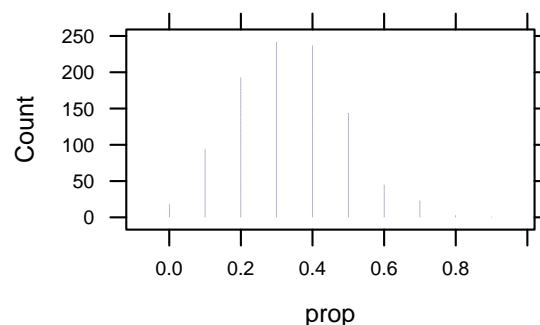
Further, we can repeat each simulation many times by multiplying it by `do()`. When using `do()`, you should assign the simulation a name by using an arrow (`<-`) so that you are creating a new data set with all of the repetitions. In this case, we are naming the simulation `GameSims`.

```
# 1000 samples, each of size 200 and proportion 1/3
GameSims <- do(1000) * rflip(n = 10, prob = 1/3)
head(GameSims)

  n heads tails prop
1 10     6     4 0.6
2 10     5     5 0.5
3 10     4     6 0.4
4 10     4     6 0.4
5 10     7     3 0.7
6 10     5     5 0.5
```

Now we can create a dotplot of the proportion of wins but note that because of there are so many observations (1000), we will not be able to see the individual dots.

```
dotPlot(~prop, data = GameSims, width = 0.1)
```



0.4 Other Visualizations

Several other types of plots can be used in place of dot plots to visualize the distribution of a single quantitative variable. The most familiar of these is the histogram, which replaces the dots of a histogram with rectangles and stacks them up touching each other to form bars. If instead we draw lines connecting the tops of each bar in a histogram (and then erase the bars), the result is a frequency polygon. A density plot is a smoother version of this idea.

Notice that to create these plots (and various numerical summaries), all we have to change is the name of the R function – all of them follow the same general template.

```
dotPlot(~ prop, data = GameSims, width = 0.1)
histogram(~ prop, data = GameSims, width = 0.1)
freqpolygon(~ prop, data = GameSims, width = 0.1, ylim=c(0,4))
densityplot(~ prop, data = GameSims)
densityplot(~ prop, data = GameSims, adjust=2) # "smoother"
densityplot(~ prop, data = GameSims, adjust=0.5) # less "smooth"
favstats(~ prop, data = GameSims)
```

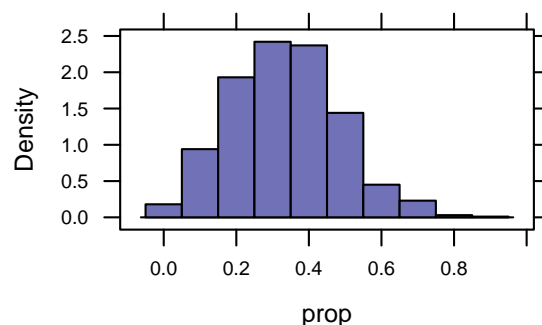
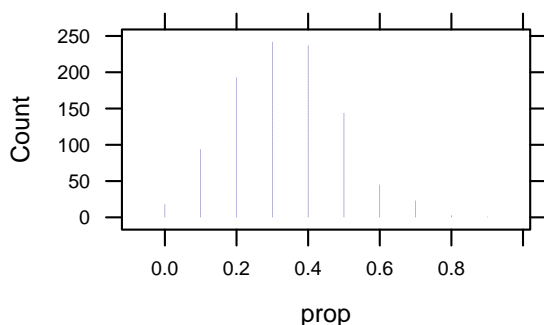
```
min  Q1 median  Q3 max   mean      sd    n missing
0 0.2   0.3 0.4 0.9 0.3338 0.1521865 1000      0
```

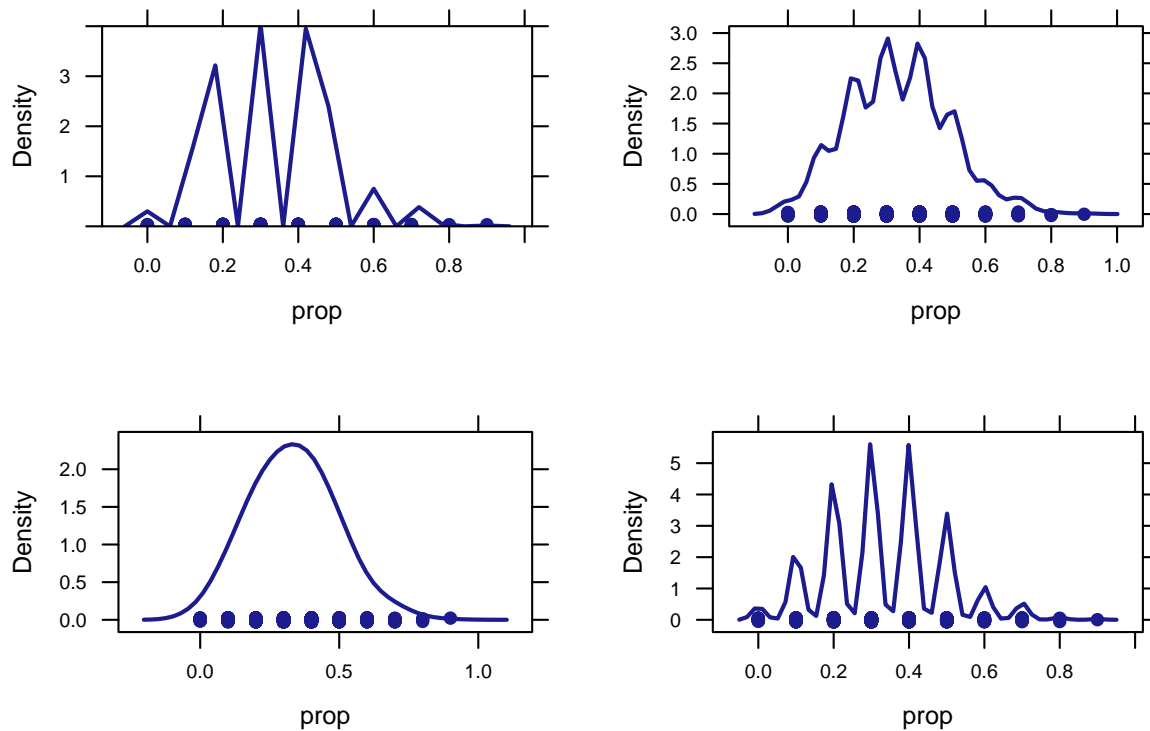
```
mean(~ prop, data = GameSims)
```

```
[1] 0.3338
```

```
sd(~ prop, data = GameSims)
```

```
[1] 0.1521865
```

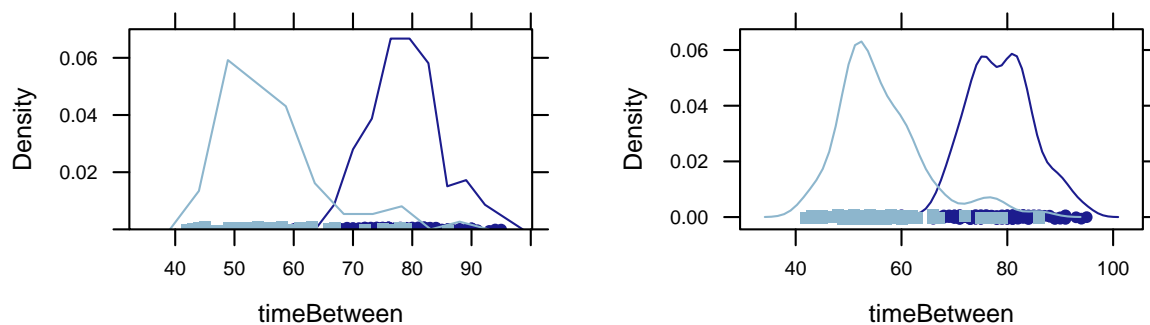




For this data set, a histogram is probably best. This is in part due to the discreteness of the data – there are only 11 possible values for `prop`.

Compared to dot plots, histograms, frequency polygons, and density plots handle a wider range of data sizes. The “sweet spot” for dot plots is around 100–1000 observations. Also, frequency polygons and density plot have the advantage that they can be overlaid.

```
freqpolygon(~timeBetween, groups = eruptionType, data = OldFaithful2, ylim = c(0, 0.07))
densityplot(~timeBetween, groups = eruptionType, data = OldFaithful2)
```



(The current version of `freqpolygon()` is not too clever about choosing the limits for the y-axis – sometimes you need to give it a hand.)

1

Significance: How strong is the evidence?

1.1 Introduction to Chance Models

Example 1.1: Can Dolphins Communicate?

The Chance Model

```
rflip(n = 16, prob = 0.5) # a sequence of 16 coin flips
```

Figure 1.2

Flipping 16 coins [Prob(Heads) = 0.5] ...

T H H T H T T T T H T T H T H

Number of Heads: 6 [Proportion Heads: 0.375]

```
rflip(n = 16, prob = 0.5) # another sequence of 16 coin flips
```

Figure 1.3

Flipping 16 coins [Prob(Heads) = 0.5] ...

T T T T H T H H T T T T H T T

Number of Heads: 4 [Proportion Heads: 0.25]

Using and evaluating the coin flip chance model

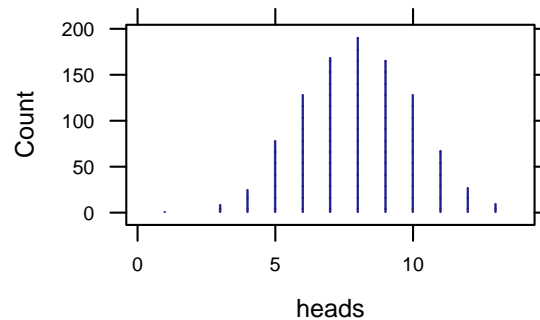
```
sim <- do(1000) * rflip(16, 0.5) # 1000 samples, each of size 16 and proportion 0.5
head(sim, 3)
```

Figure 1.4

n	heads	tails	prop
1	6	10	0.375
2	4	12	0.25

```
1 16 12 4 0.7500
2 16 11 5 0.6875
3 16 8 8 0.5000
```

```
dotPlot(~heads, data = sim, width = 1, cex = 3)
```



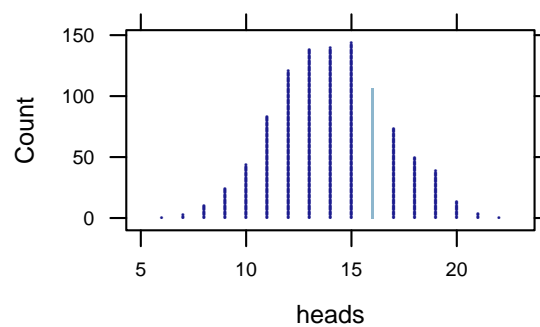
Another Doris and Buzz study

```
sim2 <- do(1000) * rflip(28, 0.5)
head(sim2, 3)
```

Figure1.6

```
  n heads tails  prop
1 28  13   15 0.4642857
2 28  13   15 0.4642857
3 28  13   15 0.4642857
```

```
dotPlot(~heads, data = sim2, width = 1, cex = 3, groups = (heads == 16))
```



Notice the way we defined `groups` as `(groups = (heads == 16))` in order to differentiate the observations where `heads` equals 16. The `==` operator means “are equal to”. (We could also have used `groups = (heads != 16)` and the colors would be reversed.)

Exploration 1.1: Can Dogs Understand Human Cues?

The Chance Model

```
sim.harley <- do(1) * rflip(10, 0.5)
sim.harley
```

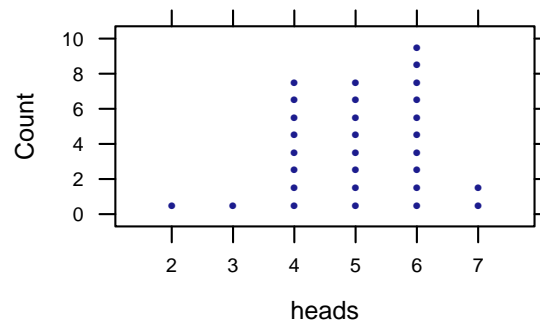
Exploration1.1.13

```
      n heads tails prop
1 10      3      7 0.3
```

```
sim.class <- do(30) * rflip(10, 0.5)
head(sim.class, 3)
```

```
      n heads tails prop
1 10      5      5 0.5
2 10      6      4 0.6
3 10      6      4 0.6
```

```
dotPlot(~heads, data = sim.class, width = 1, cex = 0.5)
```

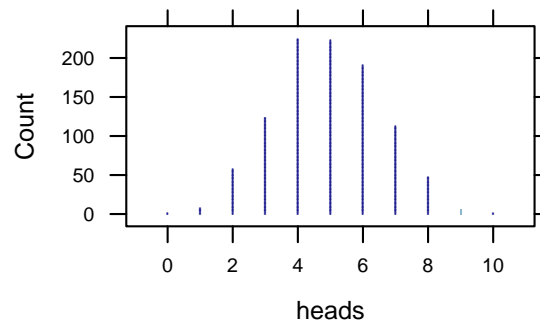


```
sim.harley2 <- do(1000) * rflip(10, 0.5)
head(sim.harley2, 3)
```

Exploration1.1.14

```
      n heads tails prop
1 10      6      4 0.6
2 10      3      7 0.3
3 10      3      7 0.3
```

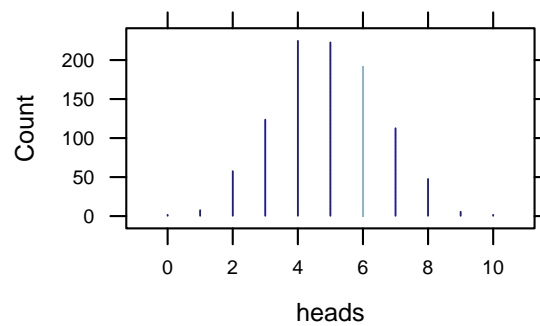
```
dotPlot(~heads, data = sim.harley2, width = 1, cex = 3, groups = (heads == 9))
```



Another Study

```
dotPlot(~heads, data = sim.harley2, width = 1, cex = 3, groups = (heads == 6))
```

Exploration1.1.23



1.2 Measuring the Strength of Evidence

Example 1.2: Rock Paper Scissors

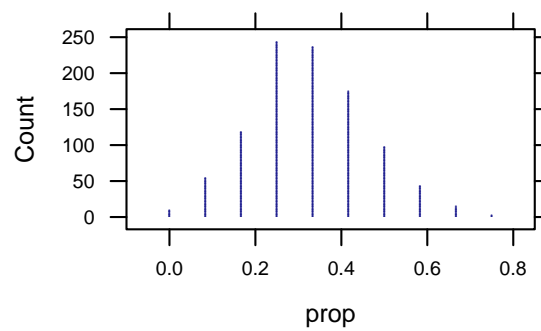
1. $H_0: \pi = 1/3$
 $H_a: \pi < 1/3$
 Test statistic: $\hat{p} = 0.167$ (the sample proportion of 1/6)
2. We simulate a world in which $\pi = 1/3$:

```
sim.sci <- do(1000) * rflip(12, 1/3)
head(sim.sci, 3)
```

Figure1.7

	n	heads	tails	prop
1	12	3	9	0.2500000
2	12	3	9	0.2500000
3	12	4	8	0.3333333

```
dotPlot(~prop, data = sim.sci, width = 1/12, cex = 3)
```

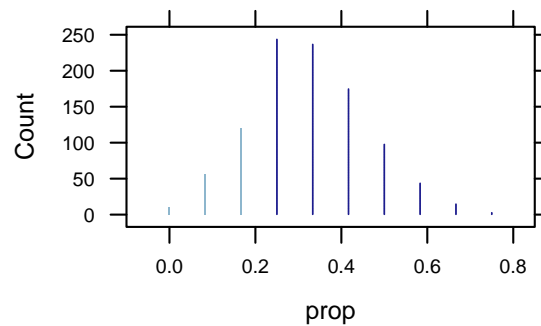
3. Strength of evidence:

For the **p-value**, you can use the `prop()` function and input `(prop <= 1/6)` to find the proportion of samples that is less than or equal to the observed proportion in the data set `sim.sci`.

```
dotPlot(~prop, data = sim.sci, cex = 3, width = 1/12, groups = (prop <= 1/6))
prop(~(prop <= 1/6), data = sim.sci)
```

Figure1.8

```
TRUE
0.184
```

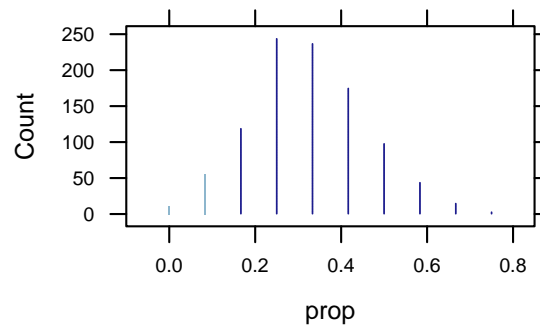


Conclusions

```
dotPlot(~prop, data = sim.sci, cex = 3, width = 1/12, groups = (prop <= 1/12))
prop(~(prop <= 1/12), data = sim.sci)
```

Figure1.9

```
TRUE
0.065
```



Exploration 1.2: Tasting Water

1. $H_0: \pi = 1/4$

$H_a: \pi < 1/4$

Test statistic: $\hat{p} = 0.111$ (the sample proportion of 3/27)

2. We simulate a world in which $\pi = 1/4$:

```
sample.tap <- do(1) * rflip(27, 1/4)
sample.tap
```

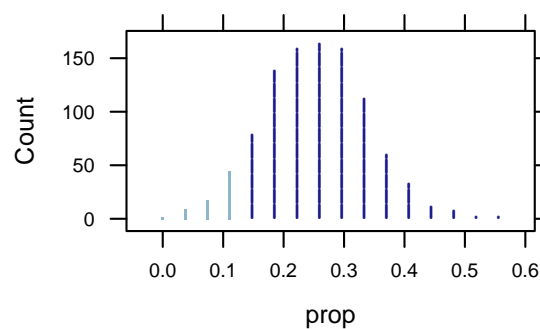
	n	heads	tails	prop
1	27	7	20	0.2592593

```
sim.tap <- do(1000) * rflip(27, 1/4)
head(sim.tap, 3)
```

	n	heads	tails	prop
1	27	11	16	0.4074074
2	27	10	17	0.3703704
3	27	6	21	0.2222222

```
dotPlot(~prop, data = sim.tap, width = 1/27, cex = 3, groups = (prop <= 3/27))
```

Exploration1.2.18



3. Strength of evidence:

```
prop(~(prop <= 3/27), data = sim.tap)
```

Exploration1.2.20

```
TRUE  
0.07
```

Alternate Analysis

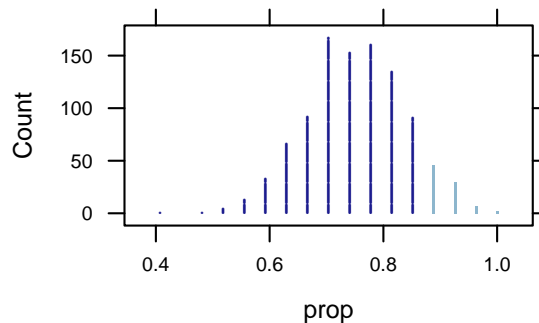
1. $H_0: \pi = 3/4$
 $H_a: \pi > 3/4$
 Test statistic: $\hat{p} = 0.889$ (the sample proportion of 24/27)
2. We simulate a world in which $\pi = 3/4$:

```
sim.bottled <- do(1000) * rflip(27, 3/4)  
head(sim.bottled, 3)
```

Exploration1.2.26

	n	heads	tails	prop
1	27	22	5	0.8148148
2	27	20	7	0.7407407
3	27	21	6	0.7777778

```
dotPlot(~prop, data = sim.bottled, width = 1/27, cex = 3, groups = (prop >= 24/27))
```



3. Strength of evidence:

```
prop(~(prop >= 24/27), data = sim.bottled)
```

Exploration1.2.26b

```
TRUE  
0.081
```

1.3 Alternative Measure of Strength of Evidence

Example 1.3: Heart Transplant Operations

1. $H_0: \pi = 0.15$

$H_a: \pi > 0.15$

Test statistic: $\hat{p} = 0.80$ (the sample proportion of 8/10)

2. We simulate a world in which $\pi = 0.15$:

```
sim.heart <- do(1000) * rflip(10, 0.15)
head(sim.heart, 3)
```

Figure1.10

```
      n heads tails prop
1 10      0     10  0.0
2 10      3      7  0.3
3 10      1      9  0.1
```

```
mean(~prop, data = sim.heart)
```

```
[1] 0.155
```

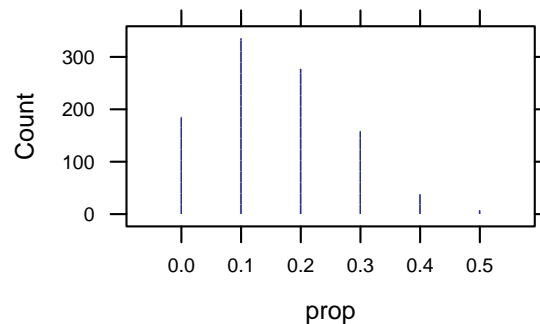
```
sd(~prop, data = sim.heart)
```

```
[1] 0.1116578
```

```
favstats(~prop, data = sim.heart)
```

```
min  Q1 median  Q3 max  mean      sd    n missing
0 0.1    0.1 0.2 0.5 0.155 0.1116578 1000      0
```

```
dotPlot(~prop, data = sim.heart, width = 0.1, cex = 3, groups = (prop >= 8/10))
```



3. Strength of evidence:

```
prop(~(prop >= 8/10), data = sim.heart)
```

Figure1.10b

```
TRUE
0
```

Digging deeper into the St. George's mortality data

1. $H_0: \pi = 0.15$

$H_a: \pi > 0.15$

Test statistic: $\hat{p} = 0.197$ (the sample proportion of 71/361)

2. We simulate a world in which $\pi = 0.15$:

```
sim.1986 <- do(1000) * rflip(361, 0.15)
head(sim.1986, 3)
```

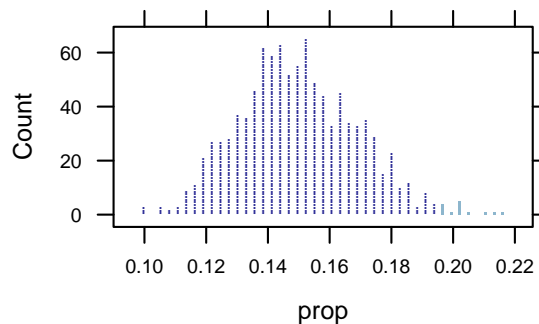
Figure1.11

```
      n heads tails      prop
1 361    56   305 0.1551247
2 361    54   307 0.1495845
3 361    44   317 0.1218837
```

```
favstats(~prop, data = sim.1986)
```

```
      min      Q1    median      Q3      max      mean      sd    n missing
0.09972299 0.1357341 0.1495845 0.1634349 0.2160665 0.1498726 0.01907189 1000      0
```

```
dotPlot(~prop, data = sim.1986, width = 1/361, groups = (prop >= 71/361))
```



3. Strength of evidence:

```
prop(~(prop >= 71/361), data = sim.1986)
```

Figure1.11b

```
TRUE
0.014
```

An alternative to the p-value: Standardized value of a statistic

R can be used as a calculator so we can compute the **z-score** manually:

```
z <- (71/361 - 0.15) / 0.018; z # z-score for sample size 361
```

```
[1] 2.593106
```

```
z <- (8/10 - 0.15) / 0.113; z # z-score for sample size 10
```

```
[1] 5.752212
```

Example1.3

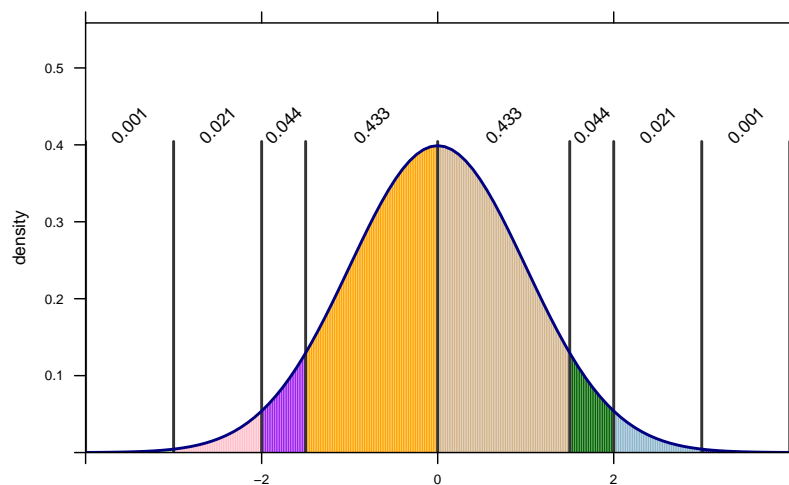
A very simple way to calculate the standardized statistic, find the p-value, and plot the bell-shaped curve is with the `xpnorm()` function. We'll examine `xpnorm()` in more detail later but for now, we just define a vector of quantiles (z-scores), `mean`, and `sd`.

```
xpnorm(c(-3, -2, -1.5, 0, 1.5, 2, 3), mean = 0, sd = 1)
```

Figure 1.12

If $X \sim N(0,1)$, then

```
P(X <= -3) = P(Z <= -3) = 0.0013
P(X <= -2) = P(Z <= -2) = 0.0228
P(X <= -1.5) = P(Z <= -1.5) = 0.0668
P(X <= 0) = P(Z <= 0) = 0.5
P(X <= 1.5) = P(Z <= 1.5) = 0.9332
P(X <= 2) = P(Z <= 2) = 0.9772
P(X <= 3) = P(Z <= 3) = 0.9987
P(X > -3) = P(Z > -3) = 0.9987
P(X > -2) = P(Z > -2) = 0.9772
P(X > -1.5) = P(Z > -1.5) = 0.9332
P(X > 0) = P(Z > 0) = 0.5
P(X > 1.5) = P(Z > 1.5) = 0.0668
P(X > 2) = P(Z > 2) = 0.0228
P(X > 3) = P(Z > 3) = 0.0013
[1] 0.001349898 0.022750132 0.066807201 0.500000000 0.933192799 0.977249868 0.998650102
```



In the example above, we input standardized values. However, we can input non-standardized statistics (observed statistic) with a new `mean` and `sd` in order to calculate the z-score.

```
xpnorm(71/361, mean = 0.15, sd = 0.018, plot = FALSE)
```

Example 1.3b

If $X \sim N(0.15, 0.018)$, then

```
P(X <= 0.196675900277008) = P(Z <= 2.593) = 0.9952
P(X > 0.196675900277008) = P(Z > 2.593) = 0.0048
```

```
[1] 0.9952443
```

```
xpnorm(8/10, mean = 0.15, sd = 0.113, plot = FALSE)
```

If $X \sim N(0.15, 0.113)$, then

$P(X \leq 0.8) = P(Z \leq 5.752) = 1$

$P(X > 0.8) = P(Z > 5.752) = 0$

```
[1] 1
```

We'll ignore the p-values and plots for now and just realize that `xpnorm()` has computed the z-score for us so that we do not need to manually compute z by using R as a calculator.

Exploration 1.3: Do People Use Facial Prototyping?

1. $H_0: \pi = 0.5$

$H_a: \pi > 0.5$

Test statistic: $\hat{p} = 0.6$ (the sample proportion of 18/30 for a fictitious class)

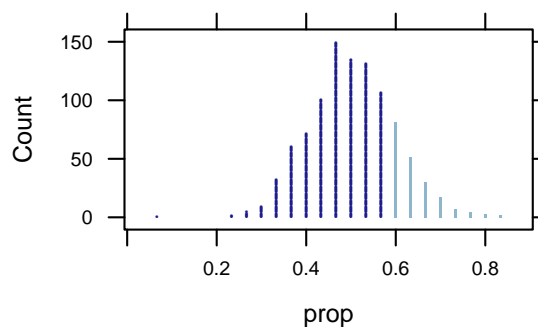
2. We simulate a world in which $\pi = 0.5$:

```
sim.tim <- do(1000) * rflip(30, 0.5)
head(sim.tim, 3)
```

Exploration1.3.7

	n	heads	tails	prop
1	30	15	15	0.5000000
2	30	16	14	0.5333333
3	30	14	16	0.4666667

```
dotPlot(~prop, data = sim.tim, width = 1/30, cex = 3, groups = (prop >= 18/30))
```



3. Strength of evidence:

```
prop(~(prop >= 18/30), data = sim.tim)
```

Exploration1.3.7b

```
TRUE
0.191
```

```
mean(~prop, data = sim.tim)
```

```
[1] 0.5013667
```

```
sd <- sd(~prop, data = sim.tim)
sd # assign the standard deviation to sd
```

```
[1] 0.09525104
```

```
z <- (0.6 - 0.5)/sd
z # z-score using the assigned sd
```

```
[1] 1.049857
```

Exploration1.3.8

Again, we can input the observed statistic, mean, and standard deviation to `xpnorm()` for the standardized statistic:

```
xpnorm(0.6, mean = 0.5, sd = sd, plot = FALSE)
```

If $X \sim N(0.5, 0.0952510395655401)$, then

$P(X \leq 0.6) = P(Z \leq 1.05) = 0.8531$

$P(X > 0.6) = P(Z > 1.05) = 0.1469$

```
[1] 0.8531081
```

Figure1.13

1.4 What Impacts Strength of Evidence?

Example 1.4: Predicting Elections from Faces?

1. $H_0: \pi = 0.5$

$H_a: \pi > 0.5$

Test statistic: $\hat{p} = 0.719$ (the sample proportion of 23/32)

2. We simulate a world in which $\pi = 0.5$:

```
sim.senate <- do(1000) * rflip(32, 0.5)
head(sim.senate, 3)
```

```
  n heads tails  prop
1 32   15    17 0.46875
2 32   17    15 0.53125
3 32   18    14 0.56250
```

```
favstats(~prop, data = sim.senate)
```

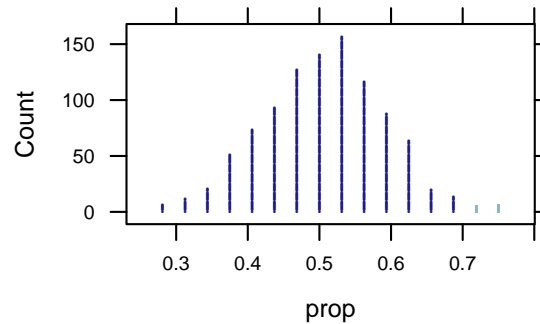
Figure1.14


```

      min      Q1 median      Q3      max      mean      sd      n missing
0.28125 0.4375    0.5 0.5625 0.75 0.5064375 0.08466908 1000      0

```

```
dotPlot(~prop, data = sim.senate, groups = (prop >= 23/32), width = 1/32, cex = 3)
```



3. Strength of evidence:

```
prop(~(prop >= 23/32), data = sim.senate)
```

Figure1.14b

```

TRUE
0.011

```

Strength of evidence with the standardized statistic:

```
mean(~prop, data = sim.senate)
```

Figure1.14c

```
[1] 0.5064375
```

```
sd <- sd(~prop, data = sim.senate)
sd
```

```
[1] 0.08466908
```

```
xpnorm(23/32, 0.5, sd, plot = FALSE)
```

If $X \sim N(0.5, 0.0846690760915471)$, then

$P(X \leq 0.71875) = P(Z \leq 2.584) = 0.9951$

$P(X > 0.71875) = P(Z > 2.584) = 0.0049$

```
[1] 0.9951111
```

What impacts strength of evidence?

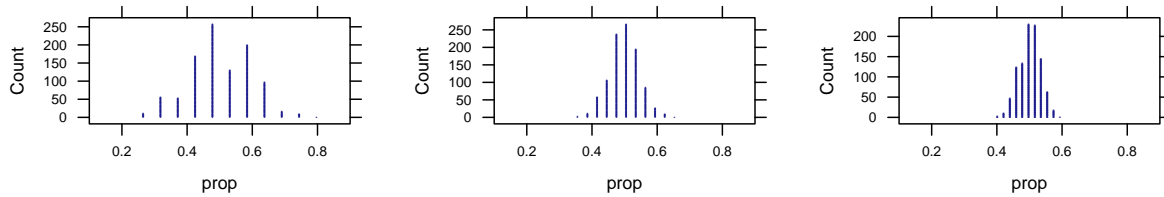
```

senate.32 <- do(1000) * rflip(32, 0.5)
dotPlot(~prop, data = senate.32, xlim = c(0.1, 0.9), cex = 5)
senate.128 <- do(1000) * rflip(128, 0.5)

```

Figure1.15

```
dotPlot(~prop, data = senate.128, xlim = c(0.1, 0.9), cex = 5)
senate.256 <- do(1000) * rflip(256, 0.5)
dotPlot(~prop, data = senate.256, xlim = c(0.1, 0.9), cex = 5)
```



```
sd(~prop, data = senate.32)
```

```
[1] 0.09114295
```

```
sd(~prop, data = senate.128)
```

```
[1] 0.04479749
```

```
sd(~prop, data = senate.256)
```

```
[1] 0.03158375
```

Figure1.15b

```
prop(~(prop >= 23/32), data = senate.32)
```

```
TRUE
0.011
```

```
prop(~(prop >= 23/32), data = senate.128)
```

```
TRUE
0
```

```
prop(~(prop >= 23/32), data = senate.256)
```

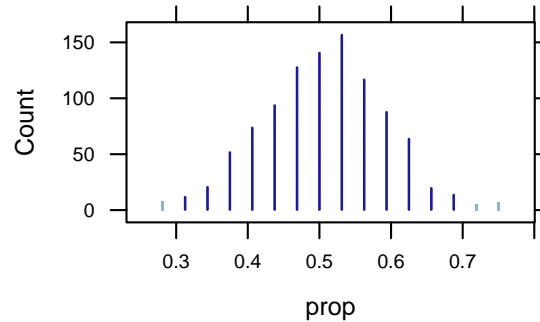
```
TRUE
0
```

Figure1.15c

1. $H_0: \pi = 0.5$
 $H_a: \pi \neq 0.5$
 Test statistic: $\hat{p} = 0.719$ (the sample proportion of 23/32)
2. We use the simulated world in which $\pi = 0.5$:

```
dotPlot(~ prop, data = sim.senate, groups = (prop >= 23/32 | prop <= 9/32),
        width = 1/32, cex = 3)
```

Figure1.16



Notice that because we are doing a two-sided test, we differentiate the samples with proportions greater than or equal to 23/32 and proportions less than or equal to 9/32 (the proportion that is as extreme as 23/32) by using the bar |.

3. Strength of evidence:

```
prop(~(prop <= 9/32 | prop >= 23/32), data = sim.senate)
```

Figure1.16b

```
TRUE
0.018
```

Follow-up Study

1. $H_0: \pi = 0.5$

$H_a: \pi \neq 0.5$

Test statistic: $\hat{p} = 0.677$ (the sample proportion of 189/279)

2. We simulate a world in which $\pi = 0.5$:

```
sim.house <- do(1000) * rflip(279, 0.5)
head(sim.house, 3)
```

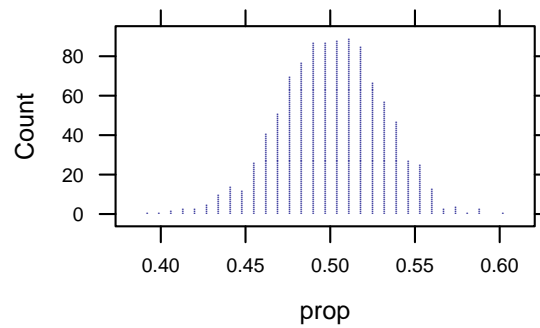
Figure1.17

```
      n heads tails      prop
1 279   146   133 0.5232975
2 279   135   144 0.4838710
3 279   124   155 0.4444444
```

```
favstats(~prop, data = sim.house)
```

```
      min      Q1    median      Q3      max      mean      sd  n missing
0.3942652 0.4802867 0.5017921 0.5197133 0.5985663 0.5002939 0.0307716 1000      0
```

```
dotPlot(~prop, data = sim.house, groups = (prop >= 189/279 | prop <= 90/279), width = 0.007)
```



3. Strength of evidence:

```
prop(~(prop >= 189/279 | prop <= 90/279), data = sim.house)
```

Figure1.17b

```
TRUE
0
```

Strength of evidence with the standardized statistic:

```
mean(~prop, data = sim.house)
```

Figure1.17c

```
[1] 0.5002939
```

```
sd <- sd(~prop, data = sim.house)
sd
```

```
[1] 0.0307716
```

```
xpnorm(189/279, 0.5, sd, plot = FALSE)
```

If $X \sim N(0.5, 0.0307716025296676)$, then

$P(X \leq 0.67741935483871) = P(Z \leq 5.766) = 1$

$P(X > 0.67741935483871) = P(Z > 5.766) = 0$

```
[1] 1
```

Exploration 1.4: Competitive Advantage to Uniform Colors?

1. $H_0: \pi = 0.5$

$H_a: \pi > 0.5$

Test statistic: $\hat{p} = 0.543$ (the sample proportion of 248/457)

2. We simulate a world in which $\pi = 0.5$:

```
sim.red <- do(1000) * rflip(457, 0.5)
head(sim.red, 3)
```

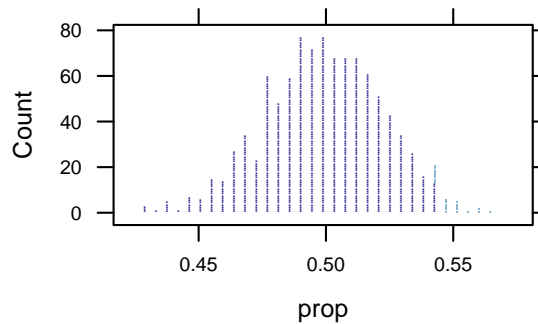
Exploration1.4.3

	n	heads	tails	prop
1	457	239	218	0.5229759
2	457	230	227	0.5032823
3	457	235	222	0.5142232

```
favstats(~prop, data = sim.red)
```

	min	Q1	median	Q3	max	mean	sd	n	missing
	0.4266958	0.4857768	0.5010941	0.5164114	0.5667396	0.5003173	0.02300691	1000	0

```
dotPlot(~prop, data = sim.red, groups = (prop >= 0.543), width = 2/457)
```



3. Strength of evidence:

```
prop(~(prop >= 0.543), data = sim.red)
```

Exploration1.4.3b

```
TRUE
0.023
```

1. $H_0: \pi = 0.5$

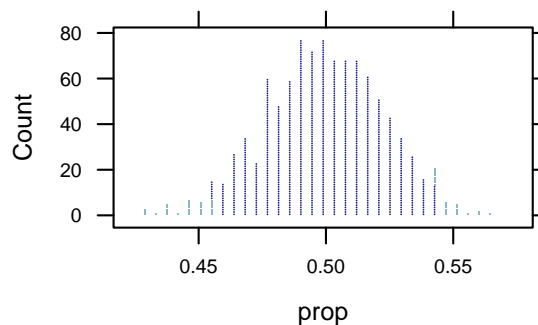
$H_a: \pi \neq 0.5$

Test statistic: $\hat{p} = 0.543$ (the sample proportion of 248/457)

2. We use the simulated world in which $\pi = 0.5$ from the one-sided test:

```
dotPlot(~prop, data = sim.red, groups = (prop <= 0.457 | prop >= 0.543), width = 2/457)
```

Exploration1.4.5



3. Strength of evidence:

```
prop(~(prop <= 0.457 | prop >= 0.543), data = sim.red)
```

Exploration1.4.5b

```
TRUE
0.053
```

Difference between statistic and null hypothesis parameter value

1. $H_0: \pi = 0.5$

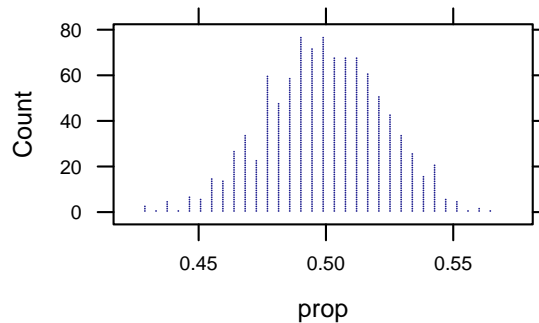
$H_a: \pi > 0.5$

Test statistic: $\hat{p} = 0.57$ (the sample proportion)

2. We use the simulated world in which $\pi = 0.5$:

```
dotPlot(~prop, data = sim.red, groups = (prop >= 0.57), width = 2/457)
```

Exploration1.4.6



3. Strength of evidence:

```
prop(~(prop >= 0.57), data = sim.red)
```

Exploration1.4.6b

```
TRUE
0
```

Sample size

1. $H_0: \pi = 0.5$

$H_a: \pi > 0.5$

Test statistic: $\hat{p} = 0.551$ (the sample proportion of 150/272)

2. We simulate a world in which $\pi = 0.5$:

```
sim.box <- do(1000) * rflip(272, 0.5)
head(sim.box, 3)
```

Exploration1.4.7

```

      n heads tails      prop
1 272  125  147 0.4595588
2 272  136  136 0.5000000
3 272  124  148 0.4558824

```

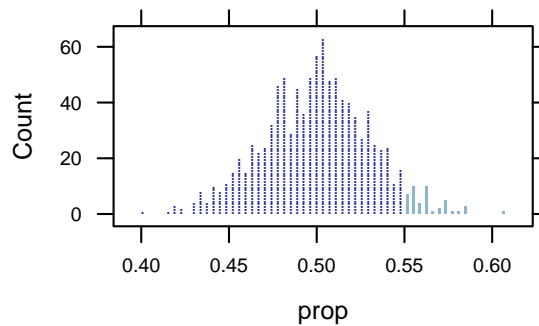
```
favstats(~prop, data = sim.box)
```

```

      min      Q1 median      Q3      max      mean      sd  n missing
0.4007353 0.4779412    0.5 0.5183824 0.6066176 0.499875 0.03002115 1000      0

```

```
dotPlot(~prop, data = sim.box, groups = (prop >= 0.551), width = 1/272)
```



3. Strength of evidence

```
prop(~(prop >= 0.551), data = sim.box)
```

Exploration 1.4.7b

```

TRUE
0.045

```

1.5 Inference on a single proportion: Theory-based approach

Example 1.5: Halloween Treats

1. $H_0: \pi = 0.5$

$H_a: \pi \neq 0.5$

Test statistic: $\hat{p} = 0.523$ (the sample proportion of 148/283)

2. We simulate a world in which $\pi = 0.5$:

```

sim.candy <- do(1000) * rflip(283, 0.5)
head(sim.candy, 3)

```

Figure 1.19

```

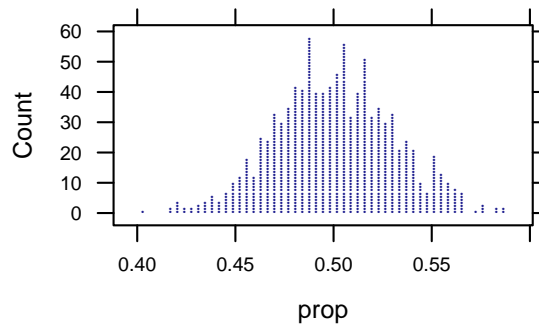
      n heads tails      prop
1 283  130  153 0.4593640
2 283  137  146 0.4840989
3 283  148  135 0.5229682

```

```
favstats(~prop, data = sim.candy)
```

	min	Q1	median	Q3	max	mean	sd	n	missing
	0.4028269	0.4805654	0.5017668	0.5194346	0.5865724	0.5004134	0.0300841	1000	0

```
dotPlot(~prop, data = sim.candy, width = 1/283)
```



Theory-based approach (One proportion z test)

Calculating predicted standard deviation:

```
mean <- 0.5
n <- 283
sd <- sqrt(mean * (1 - mean)/n)
sd
```

Example1.5

```
[1] 0.02972191
```

Calculating z-score:

```
z <- (0.523 - mean)/sd
z
```

Example1.5b

```
[1] 0.7738398
```

```
xpnorm(148/283, 0.5, sd, plot = FALSE)
```

If $X \sim N(0.5, 0.0297219149138882)$, then

$P(X \leq 0.522968197879859) = P(Z \leq 0.773) = 0.7802$

$P(X > 0.522968197879859) = P(Z > 0.773) = 0.2198$

```
[1] 0.7801707
```

To overlay a normal approximation, let's graph a histogram using `histogram()` instead of a dotplot:

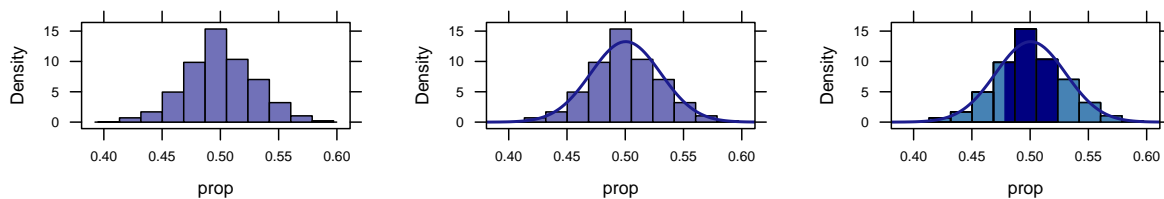

```

histogram(~prop, data = sim.candy)
histogram(~prop, data = sim.candy, fit = "normal")
histogram(~prop, data = sim.candy, fit = "normal", group = cut(prop, c(0, 135/283, 148/283,
  1)), fcol = c("steelblue", "navy", "steelblue"))
prop(~(prop <= 135/283 | prop >= 148/283), data = sim.candy)

```

Figure1.20

TRUE
0.48



The two main functions we need for working with normal distributions are `pnorm()` and `qnorm()`. `pnorm()` computes the proportion of a normal distribution below a specified value:

$$\text{pnorm}(x, \text{mean}=\mu, \text{sd}=\sigma) = \Pr(X \leq x)$$

when $X \sim \text{Norm}(\mu, \sigma)$.

We can obtain arbitrary probabilities using `pnorm()`. We can now examine the rest of the output from `xpnorm()`, which is an augmented version of `pnorm()`. Because it's a two-sided test, we can input both the observed statistic (148/283) and the statistic that is as extreme as the observed (135/283).

```
xpnorm(c(135/283, 148/283), 0.5, sd)
```

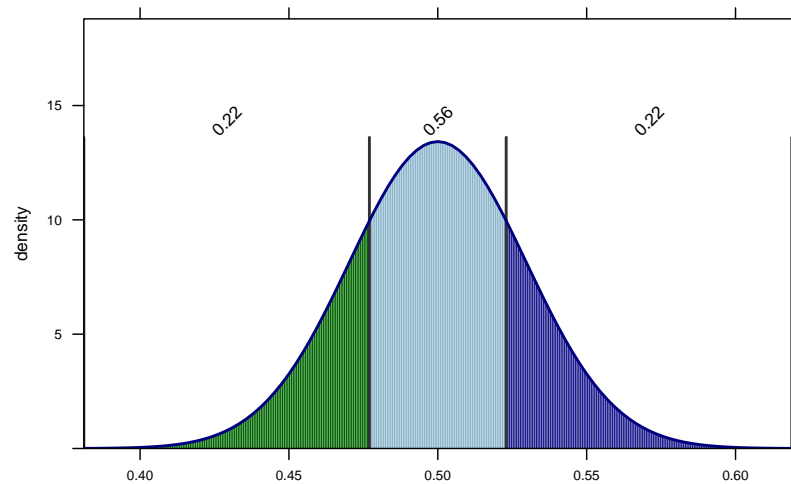
Figure1.20b

If $X \sim N(0.5, 0.0297219149138882)$, then

```

P(X <= 0.477031802120141) = P(Z <= -0.773) = 0.2198
P(X <= 0.522968197879859) = P(Z <= 0.773) = 0.7802
P(X > 0.477031802120141) = P(Z > -0.773) = 0.7802
P(X > 0.522968197879859) = P(Z > 0.773) = 0.2198
[1] 0.2198293 0.7801707

```



The output gives the z-scores for both statistics and the p-value. We know now that this p-value is found using the predicted standard deviation and normal approximation. The p-value for the two-sided test is the sum of $P(Z \leq -0.773)$ and $P(Z > 0.773)$.

We can also use the just observed statistic as we have done before but only we will need to change the `lower.tail` to `FALSE`.

```
xpnorm(148/283, 0.5, sd, lower.tail = FALSE, plot = FALSE)
```

Figure1.20c

If $X \sim N(0.5, 0.0297219149138882)$, then

$P(X \leq 0.522968197879859) = P(Z \leq 0.773) = 0.7802$

$P(X > 0.522968197879859) = P(Z > 0.773) = 0.2198$

```
[1] 0.2198293
```

```
2 * xpnorm(148/283, 0.5, sd, lower.tail = FALSE, plot = FALSE)
```

If $X \sim N(0.5, 0.0297219149138882)$, then

$P(X \leq 0.522968197879859) = P(Z \leq 0.773) = 0.7802$

$P(X > 0.522968197879859) = P(Z > 0.773) = 0.2198$

```
[1] 0.4396586
```

This results in the p-value of the alternative hypothesis that π is greater than the observed statistic (the default is the alternative hypothesis that π is less than the observed statistic). For the two-sided test, we have multiplied the resulting p-value by two.

The function `pnorm()` can be used just to find the p-value:

```
2 * pnorm(148/283, 0.5, sd, lower.tail = FALSE)
```

Figure1.20d

```
[1] 0.4396586
```

Further, we can input the standardized statistic (z-score) to find the p-value:

```
2 * pnorm(z, 0, 1, lower.tail = FALSE)

[1] 0.4390255
```

Figure1.20e

The most convenient way to find the p-value for a proportion using normal approximation is to use `prop.test()` by inputting the number of successes and the number of samples:

```
prop.test(148, n = 283)

1-sample proportions test with continuity correction

data: 148 out of 283
X-squared = 0.50883, df = 1, p-value = 0.4756
alternative hypothesis: true p is not equal to 0.5
95 percent confidence interval:
 0.4631063 0.5821963
sample estimates:
      p
0.5229682
```

Example1.5c

Note that the default for the prop test is with a $\pi = 0.5$, two-sided test, and a continuity correction. The continuity correction results in a more accurate p-value but if you want the p-value found with `pnorm()` we can change the default.

```
prop.test(148, 283, correct = FALSE)

1-sample proportions test without continuity correction

data: 148 out of 283
X-squared = 0.59717, df = 1, p-value = 0.4397
alternative hypothesis: true p is not equal to 0.5
95 percent confidence interval:
 0.4648584 0.5804628
sample estimates:
      p
0.5229682
```

Figure1.5d

A situation where a theory-based approach doesn't work

```
mean <- 1/3
n <- 12
sd <- sqrt(mean * (1 - mean)/n)
sd
```

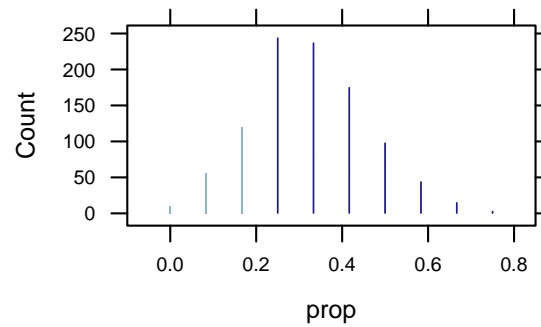
Example1.5e

```
[1] 0.1360828
```

```
dotPlot(~prop, data = sim.sci, group = (prop <= 1/6), width = 1/12, cex = 3)
prop(~(prop <= 1/6), data = sim.sci)
```

Figure1.21

```
TRUE
0.184
```



```
xpnorm(1/6, 1/3, sd)
```

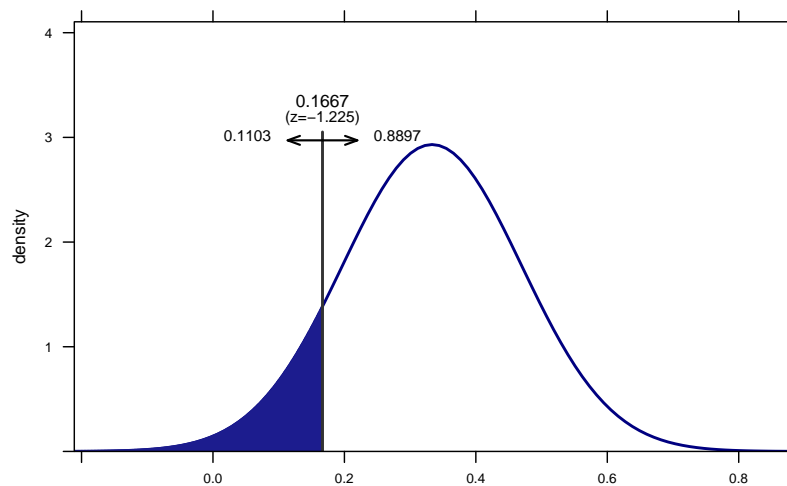
Figure1.21b

If $X \sim N(0.333333333333333, 0.136082763487954)$, then

$P(X \leq 0.166666666666667) = P(Z \leq -1.225) = 0.1103$

$P(X > 0.166666666666667) = P(Z > -1.225) = 0.8897$

```
[1] 0.1103357
```



Exploration 1.5: Calling Heads or Tails

1. $H_0: \pi = 0.5$
 $H_a: \pi > 0.5$
 Test statistic: $\hat{p} = 0.651$ (the sample proportion of 54/83)
2. We simulate a world in which $\pi = 0.5$:

```
sim.heads <- do(1000) * rflip(83, 0.5)
head(sim.heads, 3)
```

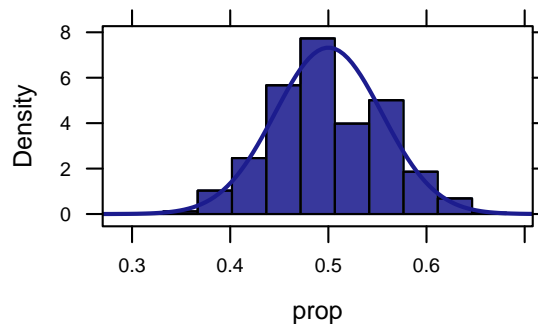
Exploration1.5.5

```
      n heads tails      prop
1 83    37    46 0.4457831
2 83    42    41 0.5060241
3 83    43    40 0.5180723
```

```
favstats(~prop, data = sim.heads)
```

```
      min      Q1    median      Q3      max      mean      sd    n missing
0.313253 0.4578313 0.4939759 0.5421687 0.6626506 0.5001084 0.05449511 1000      0
```

```
histogram(~prop, data = sim.heads, groups = (prop >= 54/83), fit = "normal")
```



3. Strength of evidence

```
prop(~(prop >= 54/83), data = sim.heads)
```

Exploration1.5.5b

```
TRUE
0.002
```

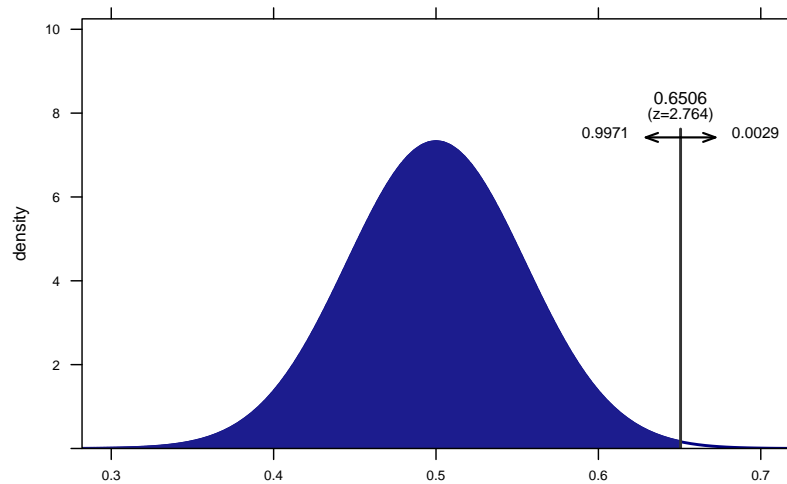
Normal approximation using simulated sd:

```
sd <- sd(~prop, data = sim.heads)
xpnorm(54/83, 0.5, sd, lower.tail = FALSE)
```

Exploration1.5.5c

If $X \sim N(0.5, 0.0544951058293514)$, then

```
P(X <= 0.650602409638554) = P(Z <= 2.764) = 0.9971
P(X > 0.650602409638554) = P(Z > 2.764) = 0.0029
[1] 0.002858421
```



Formulas

```
sd <- sqrt(0.5 * (1 - 0.5)/83)
sd
```

Exploration1.5.8

```
[1] 0.05488213
```

```
xpnorm(54/83, 0.5, sd, plot = FALSE, lower.tail = FALSE)
```

Exploration1.5.9

If $X \sim N(0.5, 0.0548821299948452)$, then

$P(X \leq 0.650602409638554) = P(Z \leq 2.744) = 0.997$

$P(X > 0.650602409638554) = P(Z > 2.744) = 0.003$

```
[1] 0.003033792
```

```
prop.test(54, 83, alt = "greater", correct = FALSE)
```

1-sample proportions test without continuity correction

data: 54 out of 83

X-squared = 7.5301, df = 1, p-value = 0.003034

alternative hypothesis: true p is greater than 0.5

95 percent confidence interval:

0.5610038 1.0000000

sample estimates:

p
0.6506024

Follow-up Analysis #1

1. $H_0: \pi = 0.5$
 $H_a: \pi \neq 0.5$
 Test statistic: $\hat{p} = 0.70$ (the sample proportion)
2. Normal approximation using predicted sd:

```
sd <- sqrt(0.5 * (1 - 0.5)/83)
sd
```

Exploration1.5.12

```
[1] 0.05488213
```

```
2 * xpnorm(0.7, 0.5, sd, plot = FALSE, lower.tail = FALSE)
```

If $X \sim N(0.5, 0.0548821299948452)$, then

$P(X \leq 0.7) = P(Z \leq 3.644) = 0.9999$

$P(X > 0.7) = P(Z > 3.644) = 1e-04$

```
[1] 0.0002682525
```

Approximate test for proportions without continuity correction:

```
prop.test(58.1, 83, correct = FALSE) # 58.1 = 0.70 * 83
```

Exploration1.5.12b

1-sample proportions test without continuity correction

data: 58.1 out of 83

X-squared = 13.28, df = 1, p-value = 0.0002683

alternative hypothesis: true p is not equal to 0.5

95 percent confidence interval:

0.5943661 0.7879397

sample estimates:

p
0.7

Follow-up Analysis # 2

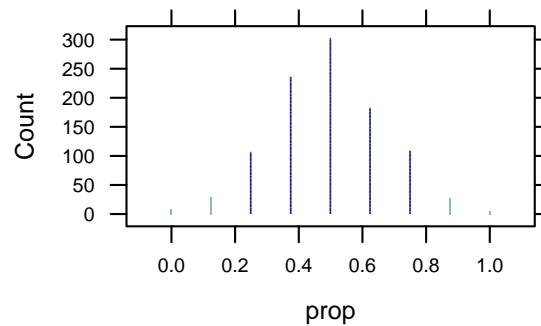
1. $H_0: \pi = 0.5$
 $H_a: \pi \neq 0.5$
 Test statistic: $\hat{p} = 0.875$ (the sample proportion of 7/8)
2. We simulate a world in which $\pi = 0.5$:

```
sim.small <- do(1000) * rflip(8, 0.5)
head(sim.small, 3)
```

Exploration1.5.13

```
  n heads tails prop
1 8     3     5 0.375
2 8     6     2 0.750
3 8     6     2 0.750
```

```
dotPlot(~prop, data = sim.small, groups = (prop <= 0.125 | prop >= 0.875), width = 1/8, cex = 3)
```



3. Strength of evidence:

```
prop(~(prop <= 0.125 | prop >= 0.875), data = sim.small)
```

Exploration1.5.13b

```
TRUE
0.065
```

Approximate test for proportions without continuity correction:

```
prop.test(7, 8, correct = FALSE)
```

Exploration1.5.13c

1-sample proportions test without continuity correction

```
data: 7 out of 8
X-squared = 4.5, df = 1, p-value = 0.03389
alternative hypothesis: true p is not equal to 0.5
95 percent confidence interval:
 0.5291118 0.9775825
sample estimates:
      p
0.875
```

There is also another test that will compute the p-value for a proportion and that the binomial test. `binom.test()` utilizes a binomial probability distribution while `prop.test()` utilizes a normal probability distribution. The tests are similar but the binomial test will result in the most accurate p-value.

```
binom.test(7, 8)
```

Exact binomial test (with Score CI)

```
data: 7 out of 8
number of successes = 7, number of trials = 8, p-value = 0.07031
alternative hypothesis: true probability of success is not equal to 0.5
95 percent confidence interval:
 0.4734903 0.9968403
sample estimates:
probability of success
      0.875
```



```
binom.test(58, 83)
```

Exact binomial test (with Score CI)

data: 58 out of 83

number of successes = 58, number of trials = 83, p-value = 0.0003783

alternative hypothesis: true probability of success is not equal to 0.5

95 percent confidence interval:

0.5882227 0.7946876

sample estimates:

probability of success

0.6987952

2

Generalization: How Broadly Do the Results Apply?

2.1 Sampling from a Finite Population

Example 2.1A: Sampling Students

```
head(CollegeMidwest, 8)
```

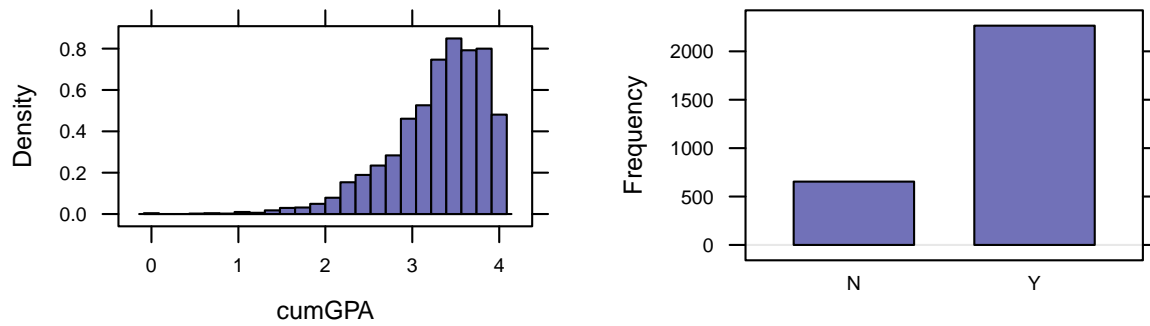
Table 2.1

	onCampus	cumGPA
1	N	2.92
2	N	3.59
3	N	3.36
4	N	2.47
5	N	3.46
6	Y	2.98
7	Y	3.07
8	Y	3.79

In chapter one, we used **histograms** a few times instead of dotplots and changed their widths. You can also control the number of bins by defining `nbint`, or `n` for short.

```
histogram(~cumGPA, data = CollegeMidwest, n = 24)
bargraph(~onCampus, data = CollegeMidwest)
```

Figure 2.1



Simple Random Samples

For a **simple random sample** of a data set, we use `sample()` and define the size of the same we want.

```
sample1 <- sample(CollegeMidwest, 30)
sample1
```

Table2.2

	onCampus	cumGPA	orig.ids
540	Y	2.72	540
1803	Y	4.00	1803
2878	Y	3.41	2878
425	N	3.63	425
1707	Y	3.79	1707
886	Y	3.81	886
2451	Y	2.85	2451
1954	N	2.38	1954
2026	Y	3.50	2026
1156	Y	3.90	1156
2714	Y	3.89	2714
1885	N	2.83	1885
832	Y	3.06	832
745	N	3.58	745
470	N	3.09	470
1650	Y	3.68	1650
1021	Y	3.52	1021
579	Y	2.84	579
2288	N	3.76	2288
1083	Y	4.00	1083
1105	Y	2.66	1105
816	Y	3.57	816
1848	Y	3.09	1848
1804	Y	2.99	1804
568	N	3.18	568
828	Y	3.33	828
193	Y	2.89	193
2245	N	3.75	2245
2212	Y	3.49	2212
2230	N	2.76	2230

```
sample2 <- sample(CollegeMidwest, 30)
sample3 <- sample(CollegeMidwest, 30)
```

```
sample4 <- sample(CollegeMidwest, 30)
sample5 <- sample(CollegeMidwest, 30)
```

Table 2.3

```
mean(~cumGPA, data = sample1)
```

```
[1] 3.331667
```

```
mean(~cumGPA, data = sample2)
```

```
[1] 3.449667
```

```
mean(~cumGPA, data = sample3)
```

```
[1] 3.325
```

```
mean(~cumGPA, data = sample4)
```

```
[1] 3.381333
```

```
mean(~cumGPA, data = sample5)
```

```
[1] 3.435667
```

```
prop(~onCampus, level = "Y", data = sample1)
```

```
Y
0.7
```

```
prop(~onCampus, level = "Y", data = sample2)
```

```
Y
0.7
```

```
prop(~onCampus, level = "Y", data = sample3)
```

```
Y
0.7333333
```

```
prop(~onCampus, level = "Y", data = sample4)
```

```
Y
0.8333333
```

```
prop(~onCampus, level = "Y", data = sample5)
```

```
Y
0.7666667
```

Notice the `level` in order to find the proportion of students who said “yes” instead of the default “no”.

Similar to the simulation of random processes in chapter one, we can repeat taking different simple random samples. Conveniently, R will let you set `data=` to a simple random sample so we can repeat finding the mean or the proportion of a different simple random sample many times.

```
sample.gpa <- do(1000) * mean(~cumGPA, data = sample(CollegeMidwest, 30))
head(sample.gpa)
```

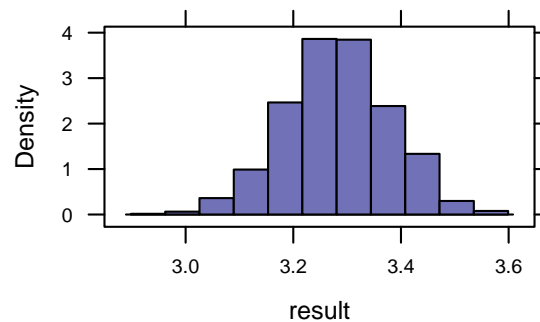
Figure2.2

```
      result
1 3.311667
2 3.388667
3 3.091333
4 3.116000
5 3.216667
6 3.238000
```

```
favstats(~result, data = sample.gpa)
```

```
      min      Q1   median      Q3     max     mean      sd  n missing
2.950667 3.2175 3.282167 3.347833 3.587667 3.282172 0.09871397 1000      0
```

```
histogram(~result, data = sample.gpa)
```



```
sample.campus <- do(1000) * prop(~onCampus, level = "Y", data = sample(CollegeMidwest, 30))
head(sample.campus)
```

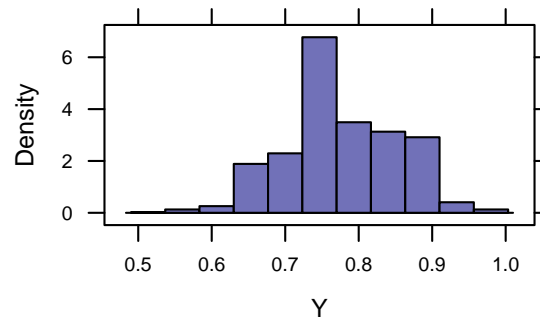
Figure2.2b

```
      Y
1 0.7666667
2 0.8333333
3 0.9000000
4 0.5666667
5 0.8000000
6 0.9000000
```

```
favstats(~Y, data = sample.campus)
```

```
min      Q1    median      Q3      max    mean      sd    n missing
0.5 0.7333333 0.7666667 0.8333333 0.9666667 0.7769 0.07483159 1000      0
```

```
histogram(~Y, data = sample.campus)
```



Exploration 2.1A: Sampling Words

```
head(GettysburgAddress)
```

```
word
1 Four
2 score
3 and
4 seven
5 years
6 ago
```

```
Words <- sample(GettysburgAddress, 10)
Words %>% mutate(length = nchar(word))
```

```
word orig.ids length
1 God 243 3
2 met 57 3
3 are 33 3
4 Now 31 3
5 nation 241 6
6 proper 97 6
7 a 246 1
8 rather 167 6
9 of 62 2
10 conceived 17 9
```

Example 2.1B: Should Supersize Drinks be Banned?

1. $H_0: \pi = 0.5$

$H_a: \pi < 0.5$

Test statistic: $\hat{p} = 0.46$ (the sample proportion of 503/1093)

2. We simulate a world in which $\pi = 0.5$:

```
sim.ban <- do(1000) * rflip(1093, 0.5)
head(sim.ban, 3)
```

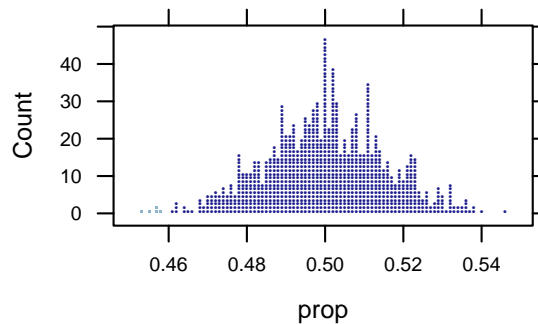
Figure2.3

```
      n heads tails      prop
1 1093   547   546 0.5004575
2 1093   560   533 0.5123513
3 1093   553   540 0.5059469
```

```
favstats(~prop, data = sim.ban)
```

```
      min      Q1   median      Q3      max      mean      sd  n missing
0.452882 0.4903934 0.5004575 0.5105215 0.5462031 0.5001372 0.01511847 1000      0
```

```
dotPlot(~prop, data = sim.ban, groups = (prop <= 0.46), width = 0.001)
```



3. Strength of evidence:

```
prop(~(prop <= 0.46), data = sim.ban)
```

Figure2.3b

```
TRUE
0.005
```

Normal approximation using predicted standard deviation:

```
sd <- sqrt(0.5 * (1 - 0.5) / 1093)
sd
```

Figure2.4

```
[1] 0.01512377
```

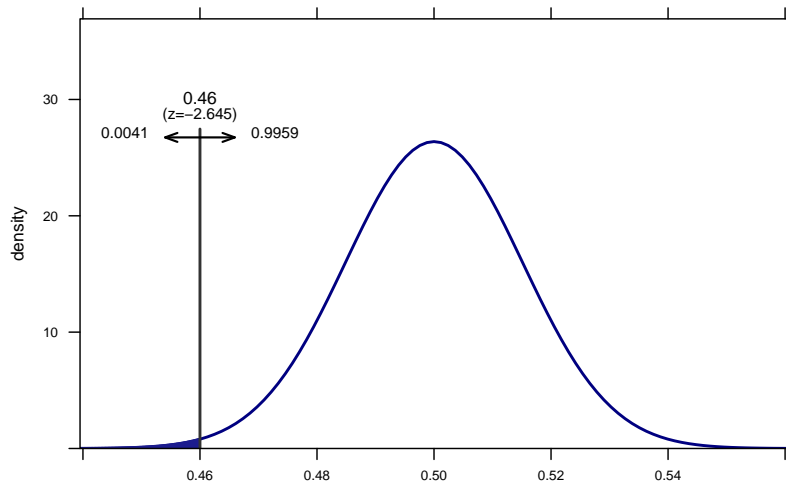
```
xpnorm(0.46, 0.5, sd)
```

```
If X ~ N(0.5, 0.0151237651004726), then
```

```
P(X <= 0.46) = P(Z <= -2.645) = 0.0041
```

```
P(X > 0.46) = P(Z > -2.645) = 0.9959
```

```
[1] 0.004086429
```

Approximate test for proportions with continuity correction:

```
prop.test(503, 1093, alt = "less")
```

Figure2.4b

1-sample proportions test with continuity correction

```
data: 503 out of 1093
X-squared = 6.7667, df = 1, p-value = 0.004644
alternative hypothesis: true p is less than 0.5
95 percent confidence interval:
 0.0000000 0.4855247
sample estimates:
          p
0.4602013
```

Exact test for proportions:

```
binom.test(503, 1093, alt = "less")
```

Figure2.4c

Exact binomial test (with Score CI)

```
data: 503 out of 1093
number of successes = 503, number of trials = 1093, p-value = 0.004628
alternative hypothesis: true probability of success is less than 0.5
95 percent confidence interval:
 0.0000000 0.4855139
sample estimates:
probability of success
 0.4602013
```

Exploration 2.1B: Banning Smoking in Cars?

1. $H_0: \pi = 0.5$
 $H_a: \pi > 0.5$

Test statistic: $\hat{p} = 0.55$ (the sample proportion)

2. We simulate a world in which $\pi = 0.5$:

```
sim.smoke <- do(1000) * rflip(1421, 0.5)
head(sim.smoke, 3)
```

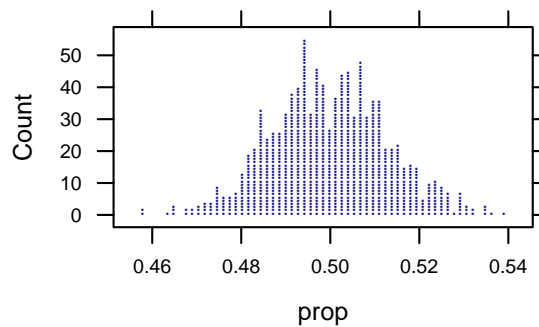
Exploration2.1B.10

```
      n heads tails      prop
1 1421   727   694 0.5116115
2 1421   706   715 0.4968332
3 1421   706   715 0.4968332
```

```
favstats(~prop, data = sim.smoke)
```

```
      min      Q1    median      Q3     max     mean      sd    n missing
0.4574243 0.4912034 0.4996481 0.5087966 0.539057 0.499912 0.01304541 1000      0
```

```
dotPlot(~prop, data = sim.smoke, groups = (prop >= 0.55), width = 0.0014)
```



3. Strength of evidence:

```
prop(~(prop >= 0.55), data = sim.smoke)
```

Exploration2.1B.10b

```
TRUE
0
```

Normal approximation using predicted standard deviation:

```
sd <- sqrt(0.5 * (1 - 0.5) / 1421)
sd
```

Exploration2.1B.14

```
[1] 0.01326395
```

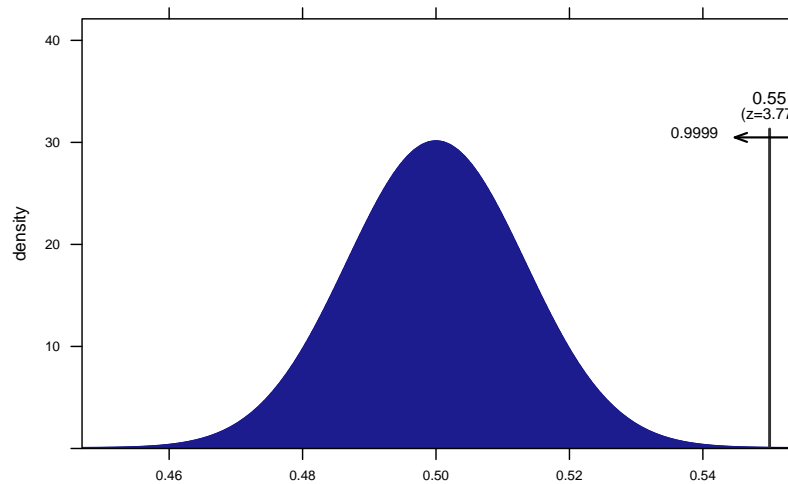
```
xpnorm(0.55, 0.5, sd, lower.tail = FALSE)
```

If $X \sim N(0.5, 0.0132639527269323)$, then

$P(X \leq 0.55) = P(Z \leq 3.77) = 0.9999$

$P(X > 0.55) = P(Z > 3.77) = 1e-04$

```
[1] 8.174966e-05
```



Approximate test for proportions with continuity correction:

```
prop.test(782, 1421, alt = "greater") # 782 = 1421 * 0.55
```

Exploration2.1B.14b

1-sample proportions test with continuity correction

```
data: 782 out of 1421
X-squared = 14.19, df = 1, p-value = 8.262e-05
alternative hypothesis: true p is greater than 0.5
95 percent confidence interval:
 0.5281822 1.0000000
sample estimates:
      p
0.5503167
```

Exact test for proportions:

```
binom.test(782, 1421, alt = "greater")
```

Exploration2.1B.14c

Exact binomial test (with Score CI)

```
data: 782 out of 1421
number of successes = 782, number of trials = 1421, p-value = 8.166e-05
alternative hypothesis: true probability of success is greater than 0.5
95 percent confidence interval:
 0.5281957 1.0000000
sample estimates:
probability of success
      0.5503167
```

2.2 Inference for a Single Quantitative Variable

Example 2.2: Estimating Elapsed Time

```
head(TimeEstimate)
```

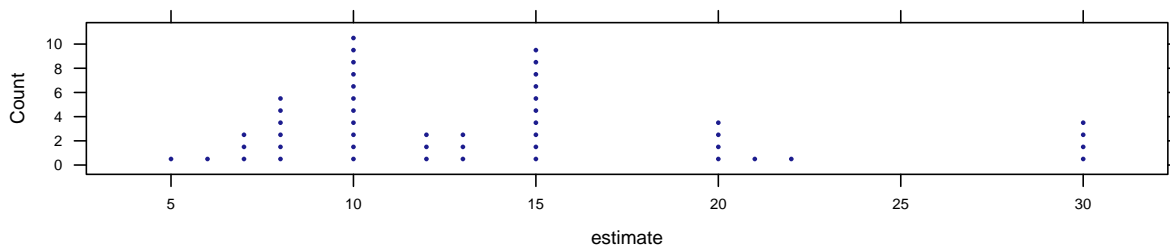
Figure 2.5

```
  estimate
1       10
2       12
3        6
4       13
5       15
6       10
```

```
favstats(~estimate, data = TimeEstimate)
```

```
min Q1 median Q3 max    mean    sd n missing
  5  10    12  15  30 13.70833 6.500273 48      0
```

```
dotPlot(~estimate, data = TimeEstimate, width = 1, cex = 0.5)
```



```
TimeEstimate %>% mutate(Rank = rank(estimate, ties.method = "random")) %>% arrange(Rank)
```

Table 2.5

```
  estimate Rank
1        5    1
2        6    2
3        7    3
4        7    4
5        7    5
6        8    6
7        8    7
8        8    8
9        8    9
10       8   10
11       8   11
12       8   12
13       8   13
14       8   14
15       8   15
16       8   16
17       8   17
18       8   18
19       8   19
20       8   20
21       8   21
22       8   22
```

23	12	23
24	12	24
25	12	25
26	13	26
27	13	27
28	13	28
29	15	29
30	15	30
31	15	31
32	15	32
33	15	33
34	15	34
35	15	35
36	15	36
37	15	37
38	15	38
39	20	39
40	20	40
41	20	41
42	20	42
43	21	43
44	22	44
45	30	45
46	30	46
47	30	47
48	30	48

```
head(TimePopulation, 3)
```

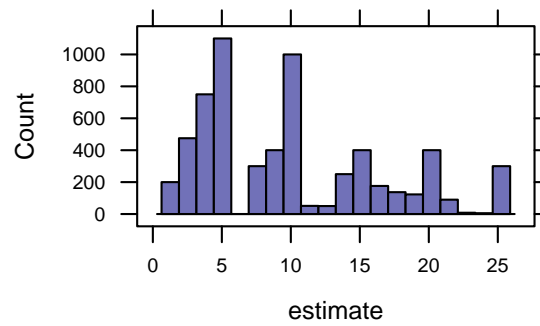
```
  estimate
1         5
2         8
3         2
```

```
favstats(~estimate, data = TimePopulation)
```

```
min Q1 median Q3 max    mean    sd    n missing
1  5      9 15  25 10.00161 6.49017 6215      0
```

```
histogram(~estimate, data = TimePopulation, type = "count", nint = 20)
```

Figure2.6



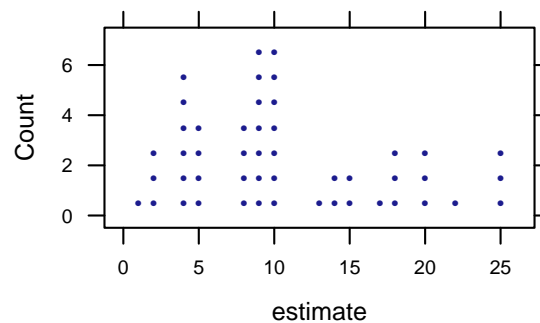
```
sample1 <- sample(TimePopulation, 48)
head(sample1, 3)
```

```
      estimate orig.ids
5599         2     5599
308         4     308
5006        20     5006
```

```
favstats(~estimate, data = sample1)
```

```
min Q1 median Q3 max      mean      sd n missing
1  5      9 15  25 10.72917 6.584119 48      0
```

```
dotPlot(~estimate, data = sample1, width = 1, cex = 0.3)
```



1. $H_0: \mu = 10$

$H_a: \mu \neq 10$

Test statistic: $\bar{x} = 13.71$ (the sample mean)

2. We simulate random samples from a finite population:

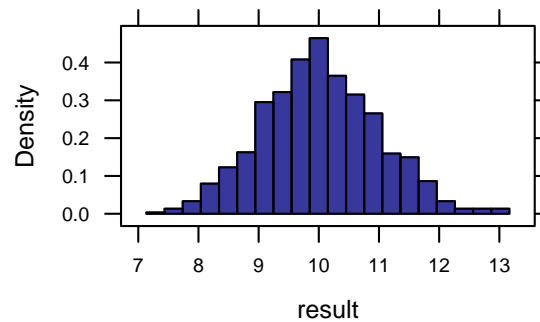
```
sim.time <- do(1000) * mean(~estimate, data = sample(TimePopulation, 48))
head(sim.time, 3)
```

```

      result
1 9.708333
2 9.625000
3 9.979167

histogram(~result, data = sim.time, groups = (result <= 6.29 | result >= 13.71), nint = 20,
          center = 10)

```



3. Strength of evidence:

```
prop(~(result <= 6.29 | result >= 13.71), data = sim.time)
```

Figure2.8b

```
TRUE
0
```

Strength of evidence with the standardized statistic:

```
mean(~result, data = sim.time)
```

Figure2.8c

```
[1] 10.02562
```

```
sd <- sd(~result, data = sim.time)
sd
```

```
[1] 0.9602245
```

```
xpnorm(13.71, 10, sd, lower.tail = FALSE, plot = FALSE)
```

If $X \sim N(10, 0.960224460941498)$, then

$P(X \leq 13.71) = P(Z \leq 3.864) = 0.9999$

$P(X > 13.71) = P(Z > 3.864) = 1e-04$

```
[1] 5.584577e-05
```

Theory-based approach: One-sample t-test

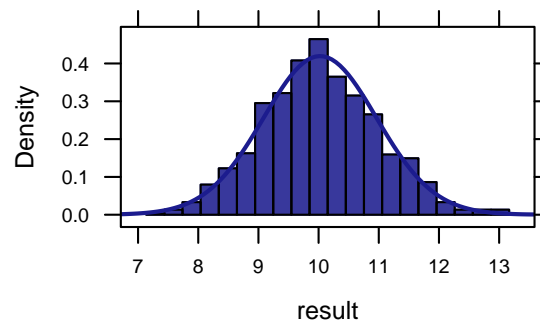
```
xbar <- 13.71
mu <- 10
s <- 6.5
n <- 48
t <- (xbar - mu)/(s/sqrt(n))
t
```

Example2.2

```
[1] 3.954405
```

```
histogram(~result, data = sim.time, groups = (result <= 6.29 | result >= 13.71), nint = 20,
  center = 10, fit = "t")
```

Figure2.9



```
2 * pt(t, df = 47, lower.tail = FALSE)
```

Figure2.10

```
[1] 0.0002570976
```

Alternative Analysis: What about the median?

```
sim.median <- do(1000) * median(~estimate, data = sample(TimePopulation, 48))
head(sim.median, 3)
```

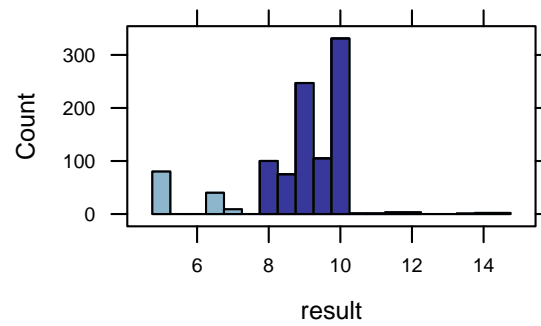
Figure2.11

```
  result
1    9.5
2    8.0
3   10.0
```

```
histogram(~result, data = sim.median, groups = (result < 8 | result > 12), width = 0.5, type = "count")
prop(~(result < 8 | result > 12), data = sim.median)
```

```
target level: TRUE; other levels: FALSE
```

```
TRUE
0.134
```

Exploration 2.2: Sleepless Nights?

```
head(SleepTimes, 3)
```

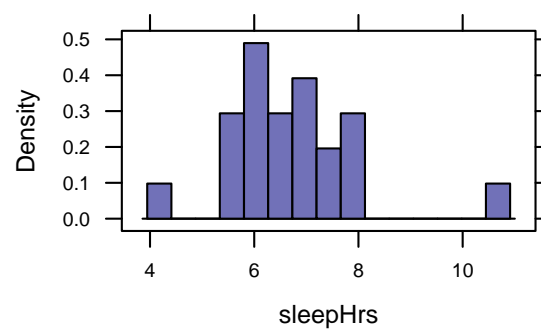
Exploration2.2.1

```
  sleepHrs  
1      7.0  
2      5.5  
3      8.0
```

Shape

```
histogram(~sleepHrs, data = SleepTimes, nint = 15)
```

Exploration2.2.10



Center

```
mean(~sleepHrs, data = SleepTimes)
```

Exploration2.2.11

```
[1] 6.704545
```

```
median(~sleepHrs, data = SleepTimes)
```

Exploration2.2.16

```
[1] 6.5
```

Variability

```
sd(~sleepHrs, data = SleepTimes)
```

Exploration2.2.18

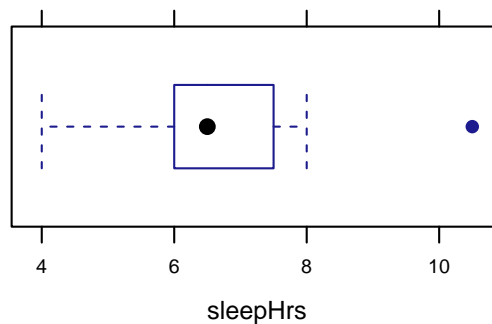
```
[1] 1.297058
```

Unusual observations

We could examine the entire data set to find any outliers, but there is a quicker way to see if there potential outliers. The `bwplot()` function plots a box-and-whisker plot which identifies *possible* outliers with a dot beyond the whiskers.

```
bwplot(~sleepHrs, data = SleepTimes)
```

Exploration2.2.20



Instead of using the hypothetical population provided in the applet, we can create our own hypothetical population by assigning a variable (`sleepHrs`) a random normal distribution (`rnorm()`) of count (18000), mean (8 hrs), and standard deviation (1.5 hrs). Additionally, let's round each value to the nearest hundredth (2) using `round()`

```
Pop1 <- data.frame(sleepHrs = round(rnorm(18000, 8, 1.5), 2))
head(Pop1)
```

Exploration2.2.24

```
  sleepHrs
1    6.10
2    7.27
3   10.84
```

```
4    5.50
5    8.90
6    6.85
```

```
favstats(sleepHrs, data = Pop1)
```

	min	Q1	median	Q3	max	mean	sd	n	missing
	2.34	6.99	8.01	9.0225	15.05	8.009642	1.50157	18000	0

```
mean(~sleepHrs, data = SleepTimes) # test statistic
```

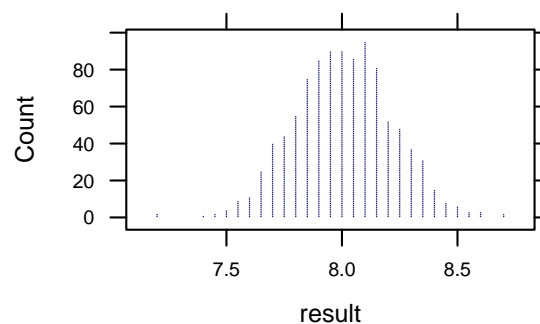
```
[1] 6.704545
```

```
sim.pop1 <- do(1000) * mean(~sleepHrs, data = sample(Pop1, 48))
head(sim.pop1, 3)
```

```
      result
1 8.248125
2 8.114792
3 7.731250
```

```
dotPlot(~result, data = sim.pop1, width = 0.05)
favstats(~result, data = sim.pop1)
```

	min	Q1	median	Q3	max	mean	sd	n	missing
	7.20125	7.862604	8.004062	8.142708	8.694583	8.006431	0.2112868	1000	0



```
prop(~(result <= 6.705), data = sim.pop1)
```

```
TRUE
0
```

```
sd <- sd(~result, data = sim.pop1)
xpnorm(6.705, 8, sd, plot = FALSE)
```

Exploration2.2.27

If $X \sim N(8, 0.211286837881223)$, then

```
P(X <= 6.705) = P(Z <= -6.129) = 0
P(X > 6.705) = P(Z > -6.129) = 1
[1] 4.418636e-10
```

```
t <- (6.705 - 8) / (1.5 / sqrt(48))
t
```

Exploration2.2.30

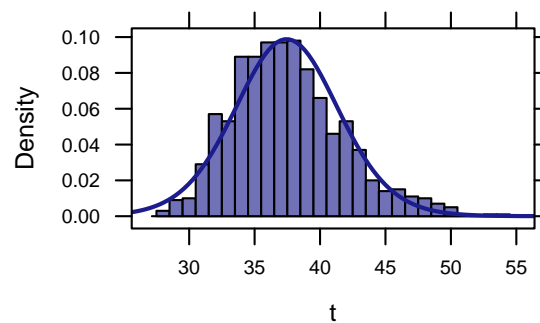
```
[1] -5.981349
```

```
sim.t <- do(1000) * stat(t.test(~sleepHrs, data = sample(Pop1, 48)))
head(sim.t, 3)
```

Exploration2.2.33

```
      t
1 36.77293
2 48.02783
3 33.31042
```

```
histogram(~t, data = sim.t, width = 1, fit = "t")
```



```
prop(~(t <= 5.981), data = sim.t)
```

Exploration2.2.34

```
TRUE
0
```

```
t.test(~sleepHrs, data = Pop1)
```

Exploration2.2.35

One Sample t-test

```
data: data$sleepHrs
t = 715.65, df = 17999, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 7.987704 8.031579
sample estimates:
mean of x
 8.009642
```

Follow-up # 1

```
head(Pop)
```

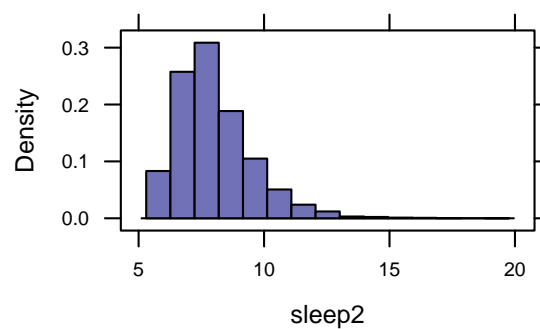
Exploration2.2.40

```
  sleep1 sleep2 sleep3
1   6.50   8.50   5.00
2   6.00  10.00   4.75
3   6.00   6.75   2.75
4   6.75  10.00   4.50
5   9.00   7.75  14.00
6   7.75   7.00  10.25
```

```
favstats(~sleep2, data = Pop)
```

```
min Q1 median  Q3 max    mean    sd    n missing
 6  7   7.75 8.75 19.5 7.999458 1.501079 18000      0
```

```
histogram(~sleep2, data = Pop)
```



```
mean(~sleepHrs, data = SleepTimes) # test statistic
```

Exploration2.2.40b

```
[1] 6.704545

sim.pop2 <- do(1000) * mean(~sleep2, data = sample(Pop, 48))
head(sim.pop2, 3)

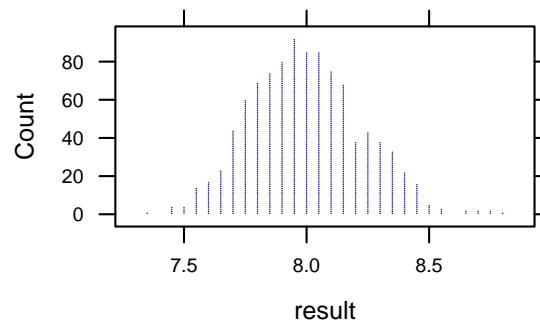
      result
1 7.947917
2 8.088542
3 7.890625

dotPlot(~result, data = sim.pop2, width = 0.05)
favstats(~result, data = sim.pop2)

      min      Q1   median      Q3      max      mean      sd      n missing
7.369792 7.838542 7.984375 8.145833 8.822917 7.997401 0.2252886 1000      0

prop(~(result <= 6.705), data = sim.pop2)

TRUE
0
```



```
t.test(~sleep2, data = Pop)
```

Exploration2.2.41

One Sample t-test

```
data: data$sleep2
t = 714.98, df = 17999, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 7.977528 8.021389
sample estimates:
mean of x
 7.999458
```

Follow-up # 2

```
median(~sleepHrs, data = SleepTimes) # test statistic
```

Exploration2.2.46

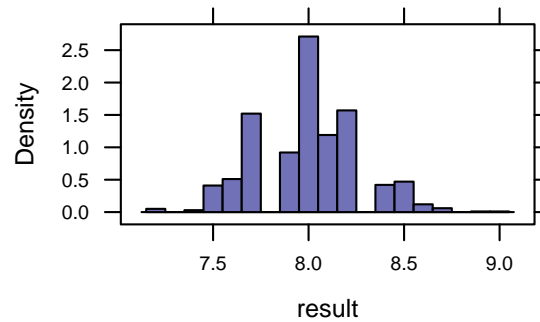
```
[1] 6.5
```

```
sim.pop1med <- do(1000) * median(~sleep1, data = sample(Pop, 48))
head(sim.pop1med, 3)
```

```
      result
1    8.25
2    8.00
3    7.75
```

```
histogram(~result, data = sim.pop1med, width = 0.1)
prop(~(result <= 6.5), data = sim.pop1med)
```

```
TRUE
0
```



2.3 Errors and Significance

Exploration 2.3: Parapsychology Studies

1. $H_0: \pi = 0.25$

$H_a: \pi > 0.25$

Test statistic: $\hat{p} = 0.333$ (the sample proportion of 709/2124)

2. We simulate a world in which $\pi = 0.25$:

```
sim.esp <- do(1000) * rflip(2124, 0.25)
head(sim.esp, 3)
```

Exploration2.3.4

```
      n heads tails      prop
1 2124   562  1562 0.2645951
2 2124   536  1588 0.2523540
3 2124   582  1542 0.2740113
```

3. Strength of evidence:

```
prop(~(prop >= 0.333), data = sim.esp)
```

Exploration2.3.4b

```
TRUE
0
```

Approximate test for proportions:

```
prop.test(709, 2124, p = 0.25, alt = "greater")
```

Exploration2.3.5

```
1-sample proportions test with continuity correction
```

```
data: 709 out of 2124
X-squared = 79.112, df = 1, p-value < 2.2e-16
alternative hypothesis: true p is greater than 0.25
95 percent confidence interval:
 0.3169623 1.0000000
sample estimates:
      p
0.3338041
```

Approximate test for $\hat{p} = 15/50$ if $\pi = 0.25$:

```
prop.test(15, 50, p = 0.25, alt = "greater")
```

Exploration2.3.12

```
1-sample proportions test with continuity correction
```

```
data: 15 out of 50
X-squared = 0.42667, df = 1, p-value = 0.2568
alternative hypothesis: true p is greater than 0.25
95 percent confidence interval:
 0.1974083 1.0000000
sample estimates:
      p
0.3
```

Approximate test for $\hat{p} = 15/50$ if $\pi = 0.33$:

```
prop.test(15, 50, p = 0.33, alt = "greater")
```

Exploration2.3.16

```
1-sample proportions test with continuity correction
```

```
data: 15 out of 50
X-squared = 0.090457, df = 1, p-value = 0.6182
alternative hypothesis: true p is greater than 0.33
95 percent confidence interval:
```



```
0.1974083 1.0000000
```

```
sample estimates:
```

```
p
```

```
0.3
```


3

Estimation: How Large is the Effect?

3.1 Statistical Inference - Confidence Intervals

Example 3.1: Can Dogs Sniff Out Cancer?

1. $H_0: \pi = 0.20$

$H_a: \pi > 0.20$

Test statistic: $\hat{p} = 0.909$ (the sample proportion of 30/33)

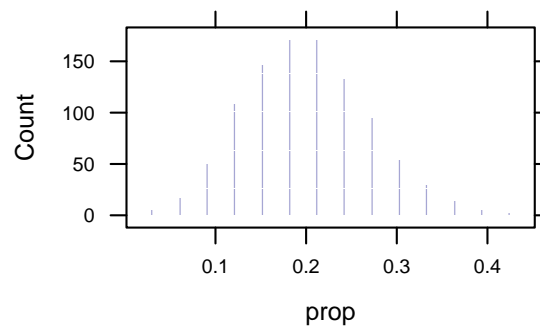
2. We simulate a world in which $\pi = 0.20$:

```
sim.cancer <- do(1000) * rflip(33, 0.2)
head(sim.cancer, 3)
```

Figure3.1

	n	heads	tails	prop
1	33	6	27	0.1818182
2	33	5	28	0.1515152
3	33	8	25	0.2424242

```
dotPlot(~prop, data = sim.cancer, groups = (prop >= 0.909), width = 0.001)
```



3. Strength of evidence:

```
favstats(~prop, data = sim.cancer)
```

Figure3.1b

	min	Q1	median	Q3	max	mean	sd	n	missing
	0.03030303	0.1515152	0.2121212	0.2424242	0.4242424	0.2003939	0.0682882	1000	0

```
prop(~(prop >= 0.909), data = sim.cancer)
```

```
TRUE
0
```

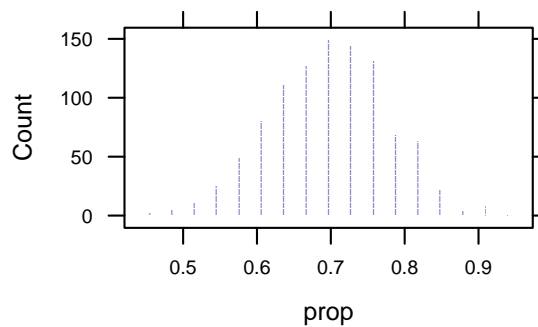
1. $H_0: \pi = 0.70$
 $H_a: \pi \neq 0.70$
 Test statistic: $\hat{p} = 0.909$ (the sample proportion of 30/33)
2. We simulate a world in which $\pi = 0.70$:

```
sim.cancer2 <- do(1000) * rflip(33, 0.7)
head(sim.cancer2, 3)
```

Figure3.2

	n	heads	tails	prop
1	33	25	8	0.7575758
2	33	18	15	0.5454545
3	33	22	11	0.6666667

```
dotPlot(~prop, data = sim.cancer2, groups = (prop <= 0.4545 | prop >= 0.909), width = 0.001)
```



3. Strength of evidence:

```
favstats(~prop, data = sim.cancer2)
```

Figure3.2b

	min	Q1	median	Q3	max	mean	sd	n	missing
	0.4545455	0.6363636	0.6969697	0.7575758	0.9393939	0.6979697	0.07967348	1000	0

```
prop(~(prop <= 0.4545 | prop >= 0.909), data = sim.cancer2)
```

```
TRUE
0.009
```

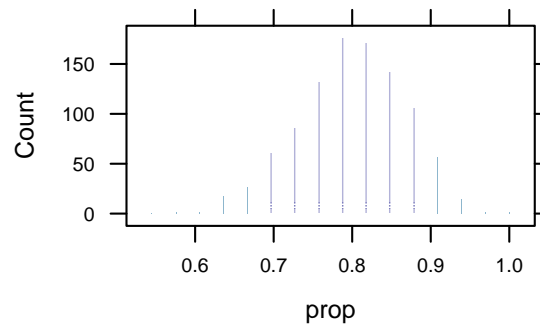
1. $H_0: \pi = 0.80$
 $H_a: \pi \neq 0.80$
 Test statistic: $\hat{p} = 0.909$ (the sample proportion of 30/33)
2. We simulate a world in which $\pi = 0.80$:

```
sim.cancer3 <- do(1000) * rflip(33, 0.8)
head(sim.cancer3, 3)
```

Figure3.3

```
      n heads tails      prop
1 33    31     2 0.9393939
2 33    25     8 0.7575758
3 33    25     8 0.7575758
```

```
dotPlot(~prop, data = sim.cancer3, groups = (prop <= 0.691 | prop >= 0.909), width = 0.001)
```



3. Strength of evidence:

```
favstats(~prop, data = sim.cancer3)
```

Figure3.3b

```
      min      Q1   median      Q3 max      mean      sd   n missing
0.5454545 0.7575758 0.7878788 0.8484848 1 0.7994848 0.06876492 1000      0
```

```
prop(~(prop <= 0.6667 | prop >= 0.909), data = sim.cancer3)
```

```
TRUE
0.126
```

Results of testing different values of probabilities under the null hypothesis:

```
pval(binom.test(30, 33, p = 0.93))
```

Table3.1

```
p.value
0.500728
```

```
pval(binom.test(30, 33, p = 0.94))
```

```

p.value
0.4474364

pval(binom.test(30, 33, p = 0.95))

p.value
0.2271931

pval(binom.test(30, 33, p = 0.96))

p.value
0.1442113

pval(binom.test(30, 33, p = 0.97))

p.value
0.0756354

pval(binom.test(30, 33, p = 0.98))

p.value
0.02792949

pval(binom.test(30, 33, p = 0.99))

p.value
0.004360339

```

Exploration 3.1: Kissing Right?

1. $H_0: \pi = 0.5$

$$H_a: \pi > 0.5$$

Test statistic: $\hat{p} = 0.645$ (the sample proportion of 80/124)

2. We simulate a world in which $\pi = 0.5$:

```

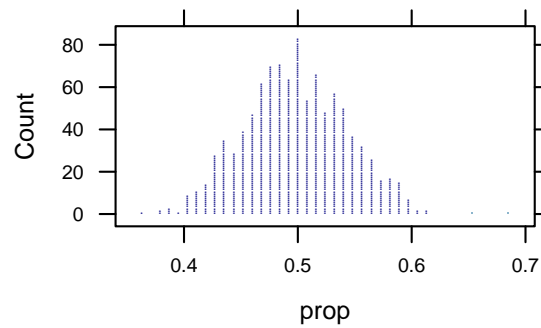
sim.kiss <- do(1000) * rflip(124, 0.5)
head(sim.kiss, 3)

      n heads tails      prop
1 124    60     64 0.4838710
2 124    61     63 0.4919355
3 124    70     54 0.5645161

dotPlot(~prop, data = sim.kiss, groups = (prop >= 0.645), width = 0.001)

```

Exploration3.1.7



3. Strength of evidence:

```
favstats(~prop, data = sim.kiss)
```

Exploration3.1.7b

	min	Q1	median	Q3	max	mean	sd	n	missing
	0.3629032	0.4677419	0.5	0.5322581	0.6854839	0.4982419	0.04482839	1000	0

```
prop(~(prop >= 0.645), data = sim.kiss)
```

```
TRUE
0.002
```

Approximate test for proportions:

```
prop.test(80, 124, alt = "greater")
```

Exploration3.1.7c

1-sample proportions test with continuity correction

```
data: 80 out of 124
X-squared = 9.879, df = 1, p-value = 0.0008359
alternative hypothesis: true p is greater than 0.5
95 percent confidence interval:
 0.5679583 1.0000000
sample estimates:
      p
0.6451613
```

Exact test for proportions:

```
binom.test(80, 124, alt = "greater")
```

Exploration3.1.7d

Exact binomial test (with Score CI)

```
data: 80 out of 124
number of successes = 80, number of trials = 124, p-value = 0.0007824
alternative hypothesis: true probability of success is greater than 0.5
95 percent confidence interval:
 0.5683679 1.0000000
sample estimates:
probability of success
      0.6451613
```

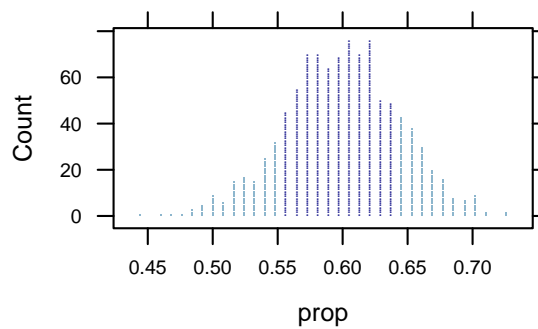
1. $H_0: \pi = 0.6$
 $H_a: \pi \neq 0.6$
 Test statistic: $\hat{p} = 0.645$ (the sample proportion of 80/124)
2. We simulate a world in which $\pi = 0.6$:

```
sim.kiss2 <- do(1000) * rflip(124, 0.6)
head(sim.kiss2, 3)
```

Exploration3.1.8

	n	heads	tails	prop
1	124	76	48	0.6129032
2	124	71	53	0.5725806
3	124	72	52	0.5806452

```
dotPlot(~prop, data = sim.kiss2, groups = (prop <= 0.555 | prop >= 0.645), width = 0.001)
```



3. Strength of evidence:

```
favstats(~prop, data = sim.kiss2)
```

Exploration3.1.8b

	min	Q1	median	Q3	max	mean	sd	n	missing
	0.4435484	0.5725806	0.5967742	0.6290323	0.7258065	0.5995565	0.04395397	1000	0

```
prop(~(prop <= 0.555 | prop >= 0.645), data = sim.kiss2)
```

```
TRUE
0.306
```

Approximate test for proportions:

```
prop.test(80, 124, p = 0.6)
```

Exploration3.1.8c

1-sample proportions test with continuity correction

```
data: 80 out of 124
X-squared = 0.87399, df = 1, p-value = 0.3499
alternative hypothesis: true p is not equal to 0.6
95 percent confidence interval:
0.5536318 0.7275562
```



```
sample estimates:
```

```
      p  
0.6451613
```

Exact test for proportions:

```
binom.test(80, 124, p = 0.6)
```

Exploration3.1.8d

Exact binomial test (with Score CI)

data: 80 out of 124

number of successes = 80, number of trials = 124, p-value = 0.3151

alternative hypothesis: true probability of success is not equal to 0.6

95 percent confidence interval:

0.5542296 0.7289832

sample estimates:

probability of success
0.6451613

```
pval(binom.test(80, 124, p = 0.54))
```

Exploration3.1.11

```
  p.value  
0.01914928
```

```
pval(binom.test(80, 124, p = 0.55))
```

```
  p.value  
0.03756733
```

```
pval(binom.test(80, 124, p = 0.56))
```

```
  p.value  
0.05778438
```

```
pval(binom.test(80, 124, p = 0.57))
```

```
  p.value  
0.1023575
```

```
pval(binom.test(80, 124, p = 0.58))
```

```
  p.value  
0.1464801
```

```
pval(binom.test(80, 124, p = 0.59))
```

```
  p.value  
0.2354593
```

```
pval(binom.test(80, 124, p = 0.6))
```

```
p.value
0.3150598
```

Exploration3.1.11b

```
pval(binom.test(80, 124, p = 0.7))
```

```
p.value
0.2023599
```

```
pval(binom.test(80, 124, p = 0.71))
```

```
p.value
0.1139799
```

```
pval(binom.test(80, 124, p = 0.72))
```

```
p.value
0.07145753
```

```
pval(binom.test(80, 124, p = 0.73))
```

```
p.value
0.04242023
```

```
pval(binom.test(80, 124, p = 0.74))
```

```
p.value
0.01849757
```

```
pval(binom.test(80, 124, p = 0.75))
```

```
p.value
0.009268747
```

```
pval(binom.test(80, 124, p = 0.76))
```

```
p.value
0.004281263
```

Exploration3.1.13

```
confint(binom.test(80, 124, p = 0.6))
```

probability of success	lower	upper
0.6451613	0.5542296	0.7289832
level		
0.9500000		

```
confint(binom.test(80, 124, p = 0.6, conf.level = 0.99))
```

Exploration3.1.15

probability of success	lower	upper
0.6451613	0.5264785	0.7523824
level		
0.9900000		

3.2 2SD and Theory-Based Confidence Intervals for a Single Proportion

Example 3.2: The Affordable Care Act

An easy way to find a confidence interval in R is to use `prop.test()` or `binom.test()` which by default calculates a 95% confidence interval in its results.

```
binom.test(713, 1034) # 713 = 1034 * 0.69
```

Example3.2

Exact binomial test (with Score CI)

```
data: 713 out of 1034
number of successes = 713, number of trials = 1034, p-value < 2.2e-16
alternative hypothesis: true probability of success is not equal to 0.5
95 percent confidence interval:
 0.6603601 0.7176665
sample estimates:
probability of success
 0.6895551
```

Theory-Based Approach

```
xpnorm(c(-1.645, 1.645), 0, 1)
```

Figure3.6

If $X \sim N(0,1)$, then

```
P(X <= -1.645) = P(Z <= -1.645) = 0.05
P(X <= 1.645) = P(Z <= 1.645) = 0.95
P(X > -1.645) = P(Z > -1.645) = 0.95
P(X > 1.645) = P(Z > 1.645) = 0.05
[1] 0.04998491 0.95001509
```

```
xpnorm(c(-1.96, 1.96), 0, 1)
```

If $X \sim N(0,1)$, then

$$P(X \leq -1.96) = P(Z \leq -1.96) = 0.025$$

$$P(X \leq 1.96) = P(Z \leq 1.96) = 0.975$$

$$P(X > -1.96) = P(Z > -1.96) = 0.975$$

$$P(X > 1.96) = P(Z > 1.96) = 0.025$$

```
[1] 0.0249979 0.9750021
```

```
xpnorm(c(-2.576, 2.576), 0, 1)
```

If $X \sim N(0,1)$, then

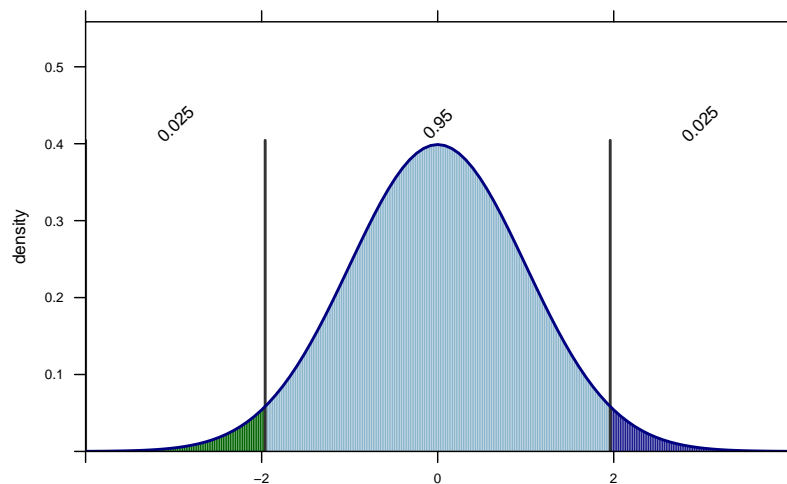
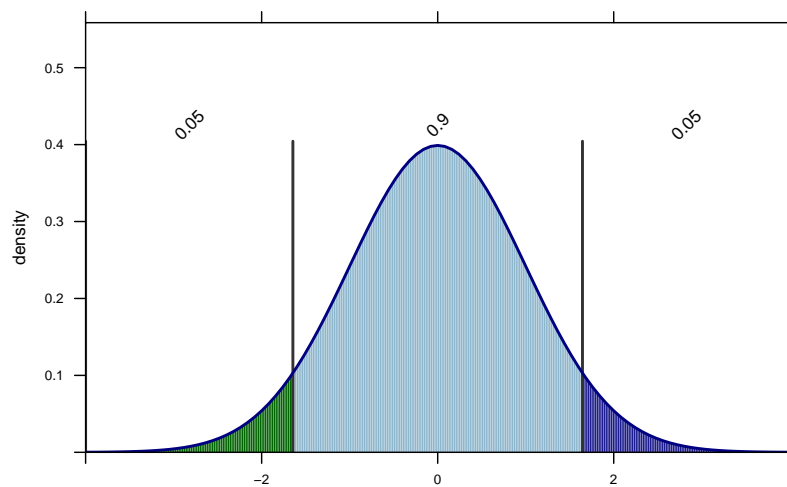
$$P(X \leq -2.576) = P(Z \leq -2.576) = 0.005$$

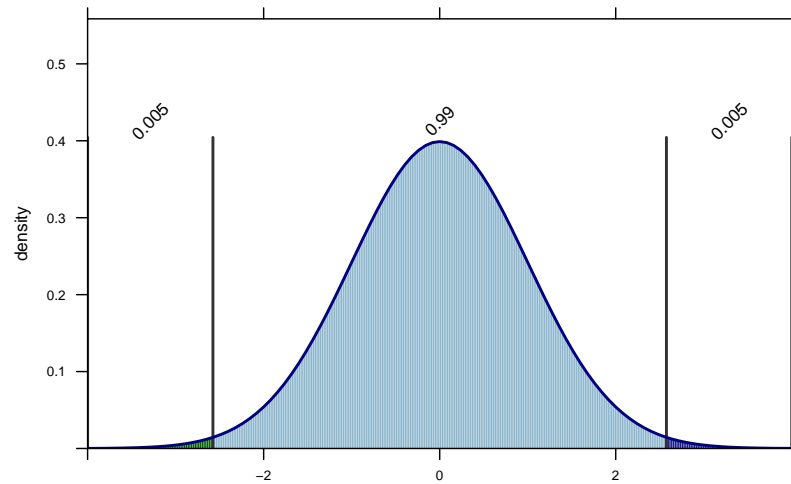
$$P(X \leq 2.576) = P(Z \leq 2.576) = 0.995$$

$$P(X > -2.576) = P(Z > -2.576) = 0.995$$

$$P(X > 2.576) = P(Z > 2.576) = 0.005$$

```
[1] 0.004997532 0.995002468
```





Using 2SD method and standard error of the observed sample proportion (Theory-Based Inference applet):

```
n <- 1034
p.hat <- 0.69; p.hat          # 0.69 = 713 / 1034

[1] 0.69

SE <- sqrt( p.hat * (1 - p.hat) / n ) # standard error
MoE <- 1.96 * SE; MoE              # margin of error

[1] 0.0281904

p.hat - MoE                      # lower limit of 95% CI

[1] 0.6618096

p.hat + MoE                      # upper limit of 95% CI

[1] 0.7181904
```

Figure3.7

Exploration 3.2: American Exceptionalism

1. $H_0: \pi = 0.775$
 $H_a: \pi \neq 0.775$
 Test statistic: $\hat{p} = 0.80$ (the sample proportion of 85/1019)
2. We simulate a world in which $\pi = 0.775$:

```
sim.amer <- do(1000) * rflip(1019, 0.775)
head(sim.amer, 3)
```

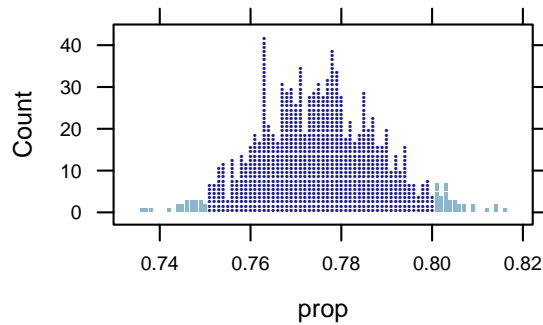
Exploration3.2.6

```

      n heads tails      prop
1 1019   817   202 0.8017664
2 1019   793   226 0.7782139
3 1019   779   240 0.7644750

```

```
dotPlot(~prop, data = sim.amer, groups = (prop <= 0.75 | prop >= 0.8), width = 0.001)
```



3. Strength of evidence:

```
favstats(~prop, data = sim.amer)
```

Exploration3.2.6b

```

      min      Q1    median      Q3      max      mean      sd    n missing
0.7360157 0.7664377 0.7752699 0.7841021 0.8155054 0.7751825 0.01311552 1000      0

```

```
prop(~(prop <= 0.75 | prop >= 0.8), data = sim.amer)
```

```

TRUE
0.056

```

Approximate test for proportions:

```
prop.test(815, 1019, p = 0.775)
```

Exploration3.2.6c

```
1-sample proportions test with continuity correction
```

```

data: 815 out of 1019
X-squared = 3.4544, df = 1, p-value = 0.06308
alternative hypothesis: true p is not equal to 0.775
95 percent confidence interval:
 0.7736183 0.8236924
sample estimates:
      p
0.7998037

```

Exact test for proportions:

```
binom.test(815, 1019, p = 0.775)
```

Exploration3.2.6d

Exact binomial test (with Score CI)

```
data: 815 out of 1019
number of successes = 815, number of trials = 1019, p-value = 0.06064
alternative hypothesis: true probability of success is not equal to 0.775
95 percent confidence interval:
 0.7738936 0.8239686
sample estimates:
probability of success
 0.7998037
```

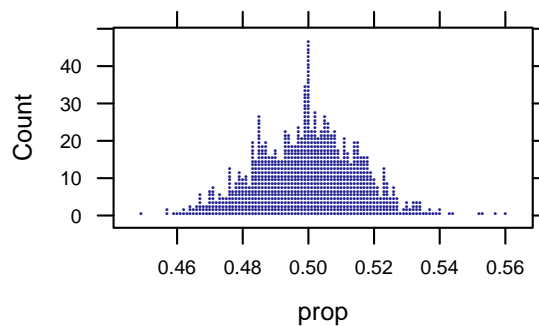
1. $H_0: \pi = 0.5$
 $H_a: \pi \neq 0.5$
 Test statistic: $\hat{p} = 0.80$ (the sample proportion of 815/1019)
2. We simulate a world in which $\pi = 0.5$:

```
sim.amer2 <- do(1000) * rflip(1019, 0.5)
head(sim.amer2, 3)
```

Exploration3.2.8

	n	heads	tails	prop
1	1019	515	504	0.5053974
2	1019	495	524	0.4857704
3	1019	494	525	0.4847890

```
dotPlot(~prop, data = sim.amer2, groups = (prop <= 0.2 | prop >= 0.8), width = 0.001)
```



3. Strength of evidence:

```
favstats(~prop, data = sim.amer2)
```

Exploration3.2.8b

min	Q1	median	Q3	max	mean	sd	n	missing
0.4494603	0.4877331	0.5004907	0.5105496	0.5603533	0.4997458	0.01606867	1000	0

```
prop(~(prop <= 0.2 | prop >= 0.8), data = sim.amer2)
```

```
TRUE
0
```

Approximate test for proportions:

```
prop.test(815, 1019)
```

Exploration3.2.8c

1-sample proportions test with continuity correction

```
data: 815 out of 1019
X-squared = 365.16, df = 1, p-value < 2.2e-16
alternative hypothesis: true p is not equal to 0.5
95 percent confidence interval:
 0.7736183 0.8236924
sample estimates:
           p
0.7998037
```

Exact test for proportions:

```
binom.test(815, 1019)
```

Exploration3.2.8d

Exact binomial test (with Score CI)

```
data: 815 out of 1019
number of successes = 815, number of trials = 1019, p-value < 2.2e-16
alternative hypothesis: true probability of success is not equal to 0.5
95 percent confidence interval:
 0.7738936 0.8239686
sample estimates:
probability of success
 0.7998037
```

Finding the standard deviation using simulated deviation:

```
sd <- sd(~prop, data = sim.amer)
sd
```

Exploration3.2.9

```
[1] 0.01311552
```

```
z <- (0.8 - 0.775)/sd
z
```

```
[1] 1.906138
```

```
xpnorm(0.8, 0.775, sd, lower.tail = FALSE, plot = FALSE)
```

If $X \sim N(0.775, 0.0131155233362715)$, then

```
P(X <= 0.8) = P(Z <= 1.906) = 0.9717
P(X > 0.8) = P(Z > 1.906) = 0.0283
[1] 0.02831614
```


Determining a 95% confidence interval using the 2SD Method and standard deviation of the null distribution:

```
p.hat <- 0.80      # given sample proportion
sd          # previously found simulated standard deviation

[1] 0.01311552

MoE <- 2 * sd; MoE  # margin of error for 95% CI

[1] 0.02623105

p.hat - MoE        # lower limit of 95% CI

[1] 0.773769

p.hat + MoE        # upper limit of 95% CI

[1] 0.826231
```

Exploration3.2.11

Determining a 95% confidence interval using the 2SD Method and standard error of the observed sample proportion:

```
n <- 1019
p.hat <- 0.80      # given sample proportion
SE <- sqrt(p.hat * (1 - p.hat) / n); SE

[1] 0.01253063

MoE <- 2 * SE; MoE  # margin of error for 95% CI

[1] 0.02506126

p.hat - MoE        # lower limit of 95% CI

[1] 0.7749387

p.hat + MoE        # upper limit of 95% CI

[1] 0.8250613
```

Exploration3.2.12

Determining a 95% confidence interval using more accurate multipliers and standard error of the observed sample proportion (Theory-Based Inference applet):

Exploration3.2.13

```

n <- 1019
p.hat <- 0.80      # given sample proportion
SE <- sqrt(p.hat * (1 - p.hat) / n); SE

[1] 0.01253063

MoE <- 1.96 * SE; MoE # margin of error for 95% CI with more accurate multiplier

[1] 0.02456003

p.hat - MoE      # lower limit of 95% CI

[1] 0.77544

p.hat + MoE      # upper limit of 95% CI

[1] 0.82456

```

Another way to create a 95% confidence interval is to use the middle 95% of the simulated null distribution. This is not exactly the same as the interval found by the 2SD Method, but it is very close.

Exploration3.2.13b

```

cdata(0.95, prop, data = sim.amer)

      low      hi central.p
0.7507360 0.8017664 0.9500000

```

The `binom.test()` calculates the exact confidence interval for any confidence level:

Exploration3.2.13c

```

binom.test(815, 1019, p = 0.775, conf.level = 0.95)

Exact binomial test (with Score CI)

data: 815 out of 1019
number of successes = 815, number of trials = 1019, p-value = 0.06064
alternative hypothesis: true probability of success is not equal to 0.775
95 percent confidence interval:
 0.7738936 0.8239686
sample estimates:
probability of success
      0.7998037

binom.test(815, 1019, p = 0.775, conf.level = 0.99)

Exact binomial test (with Score CI)

```

```
data: 815 out of 1019
number of successes = 815, number of trials = 1019, p-value = 0.06064
alternative hypothesis: true probability of success is not equal to 0.775
99 percent confidence interval:
 0.7656447 0.8311121
sample estimates:
probability of success
      0.7998037
```

```
binom.test(815, 1019, p = 0.775, conf.level = 0.9)
```

Exact binomial test (with Score CI)

```
data: 815 out of 1019
number of successes = 815, number of trials = 1019, p-value = 0.06064
alternative hypothesis: true probability of success is not equal to 0.775
90 percent confidence interval:
 0.7780614 0.8202524
sample estimates:
probability of success
      0.7998037
```

Note that the specified π , the $p = 0.775$, only matters in calculating the p-value and does not affect the confidence interval.

3.3 2SD and Theory-Based Confidence Intervals for a Single Mean

Example 3.3: Used Cars

```
head(UsedCars)
```

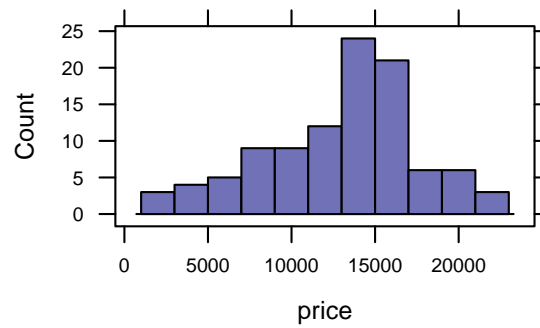
Figure 3.9

```
  price
1 21990
2 21990
3 21987
4 20955
5 20955
6 19995
```

```
favstats(~price, data = UsedCars)
```

```
   min      Q1 median      Q3    max    mean      sd  n missing
1200 10067.25 13992 15998.75 21990 13292.33 4534.568 102      0
```

```
histogram(~price, data = UsedCars, type = "count", width = 2000)
```



Determining a 95% confidence interval using the 2SD Method and standard error of the sample population:

```
n <- nrow(UsedCars); n

[1] 102

mean <- mean(~ price, data = UsedCars); mean

[1] 13292.33

sd <- sd(~ price, data = UsedCars); sd

[1] 4534.568

SE <- sd / sqrt(n)
MoE <- 2 * SE; MoE      # margin of error for 95% CI

[1] 897.9782

mean - MoE              # lower limit of 95% CI

[1] 12394.36

mean + MoE              # upper limit of 95% CI

[1] 14190.31
```

Example3.3

Theory-based approach

```
confint(t.test(~price, data = UsedCars))

mean of x      lower      upper      level
13292.33  12401.66  14183.01    0.95
```

Figure3.10

```
confint(t.test(~price, data = UsedCars, conf.level = 0.9))
```

Figure3.11

```
mean of x      lower      upper      level
13292.33 12546.98 14037.69      0.90
```

```
confint(t.test(~price, data = UsedCars, conf.level = 0.99))
```

```
mean of x      lower      upper      level
13292.33 12113.56 14471.10      0.99
```

Exploration 3.3: Sleepless Nights? (continued)

```
head(SleepTimes)
```

Exploration3.3.1

```
  sleepHrs
1      7.0
2      5.5
3      8.0
4      7.0
5      7.5
6      6.0
```

```
favstats(~sleepHrs, data = SleepTimes)
```

```
min Q1 median   Q3 max    mean      sd n missing
 4  6   6.5 7.375 10.5 6.704545 1.297058 22      0
```

Determining a 95% confidence interval using the 2SD Method and standard error of the sample population:

```
n <- nrow(SleepTimes); n
```

Exploration3.3.6

```
[1] 22
```

```
mean <- mean(~ sleepHrs, data = SleepTimes); mean
```

```
[1] 6.704545
```

```
sd <- sd(~ sleepHrs, data = SleepTimes); sd
```

```
[1] 1.297058
```

```
SE <- sd / sqrt(n)
```

```
MoE <- 2 * SE; MoE      # margin of error for 95% CI
```

```
[1] 0.5530674

mean - MoE           # lower limit of 95% CI

[1] 6.151478

mean + MoE           # upper limit of 95% CI

[1] 7.257613
```

Theory-based approach

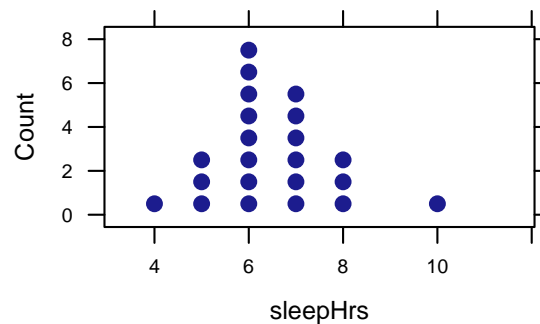
```
confint(t.test(~sleepHrs, data = SleepTimes))
```

Exploration3.3.8

mean of x	lower	upper	level
6.704545	6.129462	7.279629	0.950000

```
dotPlot(~sleepHrs, data = SleepTimes, width = 1) # to check the distribution
```

Exploration3.3.9



3.4 Factors That Affect the Width of a Confidence Interval

Example 3.4: The Affordable Care Act (continued)

```
confint(binom.test(713, 1034, conf.level = 0.9)) # 1034 * 0.69 = 713
```

Table3.5

probability of success	lower	upper
0.6895551	0.6650233	0.7132841
level		
0.9000000		

```
confint(binom.test(713, 1034, conf.level = 0.95))
```

probability of success	lower	upper
0.6895551	0.6603601	0.7176665
level		
0.9500000		

```
confint(binom.test(713, 1034, conf.level = 0.99))
```

probability of success	lower	upper
0.6895551	0.6511883	0.7261507
level		
0.9900000		

Sample size

```
confint(binom.test(70, 100))
```

Figure3.12

probability of success	lower	upper
0.7000000	0.6001853	0.7875936
level		
0.9500000		

```
confint(binom.test(140, 200))
```

probability of success	lower	upper
0.7000000	0.6313501	0.7626104
level		
0.9500000		

```
confint(binom.test(280, 400))
```

probability of success	lower	upper
0.7000000	0.6524781	0.7445333
level		
0.9500000		

Optional: Effect of sample proportion

Sample proportions will affect confidence intervals calculated by using accurate multipliers and the standard error of the observed sample proportion (Theory-Based Inference applet). However, the sample proportions will not affect confidence intervals found by using the exact test for proportions, `binom.test()`.

```
confint(binom.test(838, 1034))
```

Figure3.13

probability of success	lower	upper

```

0.8104449      0.7852004      0.8339078
level
0.9500000

MoE838 <- 0.8339078 - 0.7852004
MoE838

[1] 0.0487074

confint(binom.test(196, 1034))

probability of success      lower      upper
0.1895551      0.1660922      0.2147996
level
0.9500000

MoE196 <- 0.2147996 - 0.1660922
MoE196

[1] 0.0487074

```

Exploration 3.4: Holiday Spending Habits

Determining a 95% confidence interval using the 2SD Method and standard error of the sample population:

```

n <- 1039
mean <- 704
sd <- 150
SE <- sd / sqrt(n)
MoE <- 2 * SE; MoE      # margin of error for 95% CI

[1] 9.307081

mean - MoE      # lower limit of 95% CI

[1] 694.6929

mean + MoE      # upper limit of 95% CI

[1] 713.3071

```

Exploration3.4.5

```

n <- 1039
mean <- 704
sd <- 300
SE <- sd / sqrt(n)
MoE <- 2 * SE; MoE      # margin of error for 95% CI

```

Exploration3.4.6


```
[1] 18.61416

mean - MoE           # lower limit of 95% CI

[1] 685.3858

mean + MoE           # upper limit of 95% CI

[1] 722.6142
```

The impact of sample size

```
n <- 477
mean <- 704
sd <- 300
SE <- sd / sqrt(n)
MoE <- 2 * SE; MoE      # margin of error for 95% CI

[1] 27.47211

mean - MoE           # lower limit of 95% CI

[1] 676.5279

mean + MoE           # upper limit of 95% CI

[1] 731.4721
```

Exploration3.4.8

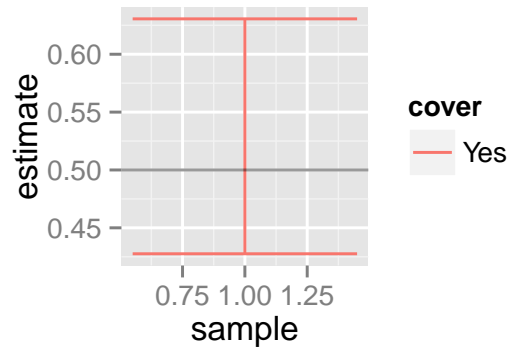
Exploration 3.4B: Reese's Pieces

Simulate 1 sample proportion and calculate the 95% confidence interval:

```
sample.CI <- CIsim(100, samples = 1, rdist = rbinom, args = list(size = 1, prob = 0.5), method = binom.test,
  method.args = list(success = 1), verbose = FALSE, estimand = 0.5)
sample.CI
```

	lower	upper	estimate	cover	sample
1	0.4275815	0.6305948	0.53	Yes	1

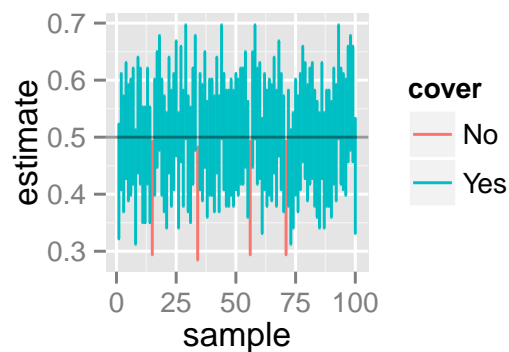
Exploration3.4B.4



Simulate 100 sample proportions and calculate the 95% confidence intervals:

```
sim.CI <- CIsim(100, samples = 100, rdist = rbinom, args = list(size = 1, prob = 0.5), method = binom.test,
  method.args = list(success = 1), verbose = FALSE, estimand = 0.5)
```

Exploration3.4B.5



Proportion of intervals produced that do not contain $\pi = 0.5$:

```
prop(~cover, data = sim.CI)
```

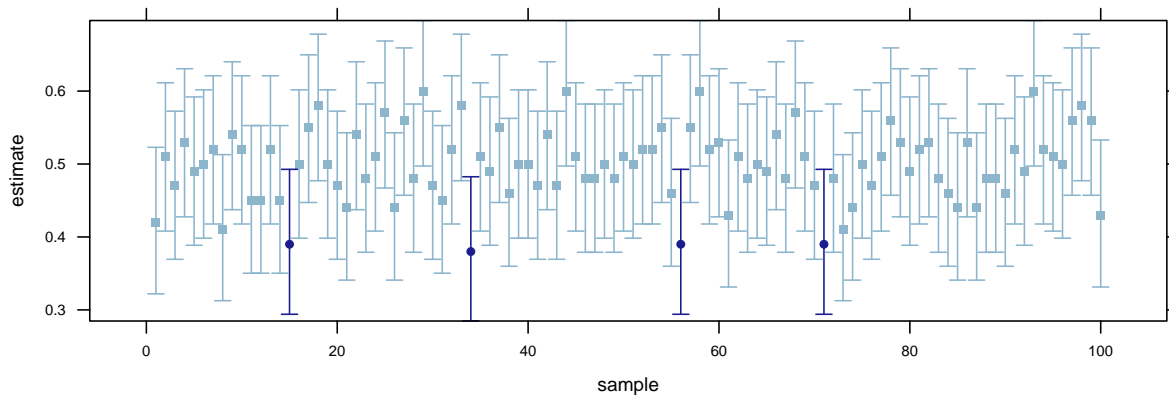
Exploration3.4B.5b

```
No
0.04
```

Plot the 95% confidence intervals of the simulation of 100 sample proportions:

```
require(Hmisc)
xyplot(Cbind(estimate, lower, upper) ~ sample, data = sim.CI, par.settings = col.mosaic(),
  groups = cover)
```

Exploration3.4B.5c



Simulate 1000 sample proportions and calculate the 95% confidence intervals:

```
sim.CI2 <- CIsim(100, samples = 1000, rdist = rbinom,
  args = list(size = 1, prob = 0.5), method = binom.test,
  method.args = list(success = 1), verbose = FALSE,
  estimand = 0.5)
```

Exploration3.4B.5d

Proportion of intervals produced that do not contain $\pi = 0.5$:

```
prop(~cover, data = sim.CI2)
```

Exploration3.4B.5e

```
No
0.04
```

Simulate 1000 sample proportions and calculate the 90% confidence intervals:

```
sim.CI3 <- CIsim(100, samples = 1000, rdist = rbinom,
  args = list(size = 1, prob = 0.5), conf.level = 0.90,
  method = binom.test, method.args = list(success = 1),
  verbose = FALSE, estimand = 0.5)
```

Exploration3.4B.6

Proportion of intervals produced that do not contain $\pi = 0.5$:

```
prop(~cover, data = sim.CI3)
```

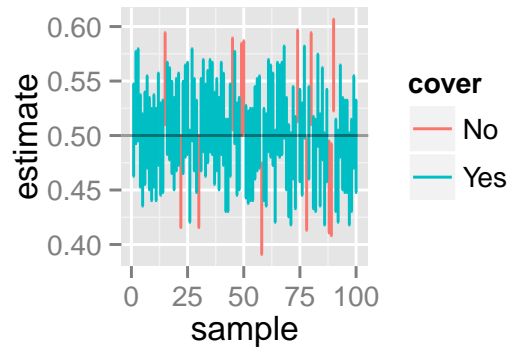
Exploration3.4B.6b

```
No
0.101
```

Simulate 1000 sample proportions and calculate the 90% confidence intervals (sample size = 400):

```
sim.CI4 <- CIsim(400, samples = 100, rdist = rbinom,
  args = list(size = 1, prob = 0.5), conf.level = 0.90,
  method = binom.test, method.args = list(success = 1),
  verbose = FALSE, estimand = 0.5)
```

Exploration3.4B.6c



Proportion of intervals produced that do not contain $\pi = 0.5$:

```
prop(~cover, data = sim.CI4)
```

Exploration3.4B.6d

```
No
0.13
```

3.5 Cautions When Conducting Inference

1. $H_0: \pi = 0.3645$

$H_a: \pi > 0.3645$

Test statistic: $\hat{p} = 0.41$ (the sample proportion)

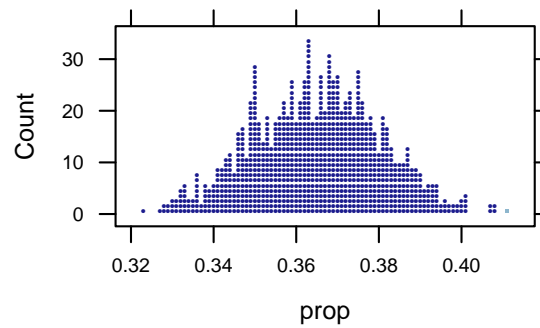
2. We simulate a world in which $\pi = 0.3645$:

```
sim.obama <- do(1000) * rflip(1000, 0.3645)
head(sim.obama, 3)
```

Figure3.14

```
      n heads tails prop
1 1000   365   635 0.365
2 1000   376   624 0.376
3 1000   375   625 0.375
```

```
dotPlot(~prop, data = sim.obama, groups = (prop >= 0.41), width = 0.001)
```



3. Strength of evidence:

```
favstats(~prop, data = sim.obama)
```

Figure3.14b

min	Q1	median	Q3	max	mean	sd	n	missing
0.323	0.353	0.365	0.376	0.411	0.364884	0.01545605	1000	0

```
prop(~(prop >= 0.41), data = sim.obama)
```

```
TRUE
0.001
```

Exploration 3.5A: Voting for President

Finding the 99% confidence interval using the exact test for proportions:

```
confint(binom.test(1783, 2613, conf.level = 0.99))
```

Exploration3.5A.3

probability of success	lower	upper
0.6823574	0.6583871	0.7056569
level		
0.9900000		

Another famous case of problems in Presidential election polling

Finding the 99% confidence interval using the exact test for proportions:

```
confint(binom.test(1368000, 2400000, conf.level = 0.99)) # 1368000 = 2400000 * 0.57
```

Exploration3.5A.9

probability of success	lower	upper
0.5700000	0.5689480	0.5710515
level		
0.9990000		

Example 3.5B: Parapsychology Studies (continued)

```
confint(binom.test(709, 2124, conf.level = 0.95))
```

Example3.5B

probability of success	lower	upper
0.3338041	0.3137548	0.3543132
level		
0.9500000		

```
confint(binom.test(709, 2124, conf.level = 0.99))
```

probability of success	lower	upper
0.3338041	0.3076114	0.3607496
level		
0.9900000		

1. $H_0: \pi = 0.25$

$H_a: \pi > 0.25$

Test statistic: $\hat{p} = 0.38$ (the sample proportion of 19/50)

2. We simulate a world in which $\pi = 0.25$:

```
sim.esp2 <- do(10000) * rflip(50, 0.25)
head(sim.esp2, 3)
```

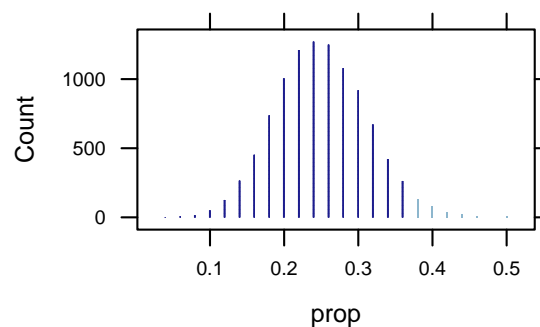
Figure3.15

	n	heads	tails	prop
1	50	10	40	0.20
2	50	12	38	0.24
3	50	15	35	0.30

```
dotPlot(~prop, data = sim.esp2, groups = (prop >= 0.38), width = 0.01, cex = 10)
prop(~(prop >= 0.38), data = sim.esp2)
```

target level: TRUE; other levels: FALSE

TRUE
0.0257



1. $H_0: \pi = 1/3$
 $H_a: \pi > 1/3$
 Test statistic: $\hat{p} = 0.38$ (the sample proportion of 19/50)
2. We simulate a world in which $\pi = 1/3$:

```
sim.esp3 <- do(10000) * rflip(50, 1/3)
head(sim.esp3, 3)
```

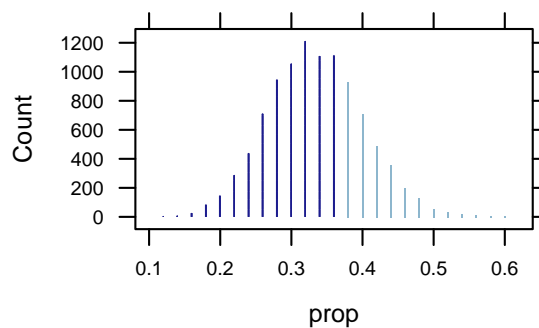
Figure3.16

	n	heads	tails	prop
1	50	19	31	0.38
2	50	13	37	0.26
3	50	23	27	0.46

```
dotPlot(~prop, data = sim.esp3, groups = (prop >= 0.38), width = 0.01, cex = 10)
prop(~(prop >= 0.38), data = sim.esp3)
```

target level: TRUE; other levels: FALSE

TRUE
0.2872



1. $H_0: \pi = 1/2$
 $H_a: \pi > 1/2$
 Test statistic: $\hat{p} = 0.38$ (the sample proportion of 19/50)
2. We simulate a world in which $\pi = 1/2$:

```
sim.esp4 <- do(10000) * rflip(50, 1/2)
head(sim.esp4, 3)
```

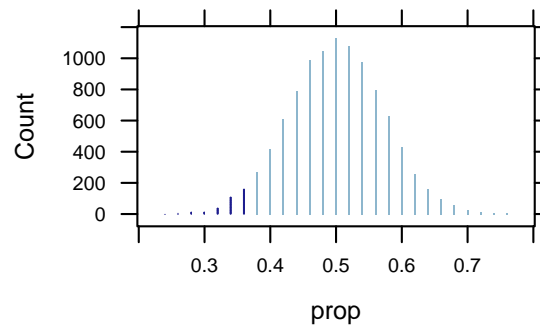
Figure3.17

	n	heads	tails	prop
1	50	27	23	0.54
2	50	30	20	0.60
3	50	29	21	0.58

```
dotPlot(~prop, data = sim.esp4, groups = (prop >= 0.38), width = 0.01, cex = 10)
prop(~(prop >= 0.38), data = sim.esp4)
```

target level: TRUE; other levels: FALSE

TRUE
0.9664



3.5.1 Exploration 3.5B: Cat Households

1. $H_0: \pi = 1/3$

$$H_a: \pi < 1/3$$

Test statistic: $\hat{p} = 0.324$ (the sample proportion of 15228/47000)

2. Exact test for proportions:

```
binom.test(15228, 47000, p = 1/3, conf.level = 0.999, alt = "less")
```

Exploration3.5B.3

Exact binomial test (with Score CI)

data: 15228 out of 47000

number of successes = 15228, number of trials = 47000, p-value = 8.654e-06

alternative hypothesis: true probability of success is less than 0.3333333

99.9 percent confidence interval:

0.0000000 0.3307064

sample estimates:

probability of success

0.324

```
binom.test(15228, 47000, p = 1/3, alt = "less")
```

Exact binomial test (with Score CI)

data: 15228 out of 47000

number of successes = 15228, number of trials = 47000, p-value = 8.654e-06

alternative hypothesis: true probability of success is less than 0.3333333

95 percent confidence interval:

0.0000000 0.3275694

sample estimates:

probability of success

0.324

3. We simulate a world in which $\pi = 1/3$:

```
sim.pets <- do(1000) * rflip(100, 1/3)
head(sim.pets, 3)
```

Exploration3.5B.9

	n	heads	tails	prop
1	100	29	71	0.29
2	100	33	67	0.33
3	100	40	60	0.40

We could use trial-and-error to determine values of the sample proportion that would produce a p-value of less than 0.05. R can quickly calculate try possible values that would result in the significance level of 0.05 but we can also have R calculate them for us.

```
cdata(0.95, prop, data = sim.pets)
```

Exploration3.5B.9b

low	hi	central.p
0.25	0.42	0.95

1. $H_0: \pi = 0.30$

$H_a: \pi < 0.30$

Test statistic: $\hat{p} = 0.243$ (the sample proportion)

2. We simulate a world in which $\pi = 0.30$:

```
sim.pets2 <- do(1000) * rflip(100, 0.3)
head(sim.pets2, 3)
```

Exploration3.5B.11

	n	heads	tails	prop
1	100	31	69	0.31
2	100	36	64	0.36
3	100	28	72	0.28

```
prop(!(prop <= 0.243), data = sim.pets2)
```

target level: TRUE; other levels: FALSE

TRUE
0.093

```
cdata(0.9, prop, data = sim.pets2)
```

Exploration3.5B.11b

low	hi	central.p
0.23	0.38	0.90

```
confint(binom.test(33, 100, p = 1/3))
```

probability of success	lower	upper
0.3300000	0.2391985	0.4311728
level		
0.9500000		

```
binom.test(24, 100, p = 0.3, alt = "less")
```

Exact binomial test (with Score CI)

data: 24 out of 100
 number of successes = 24, number of trials = 100, p-value = 0.1136
 alternative hypothesis: true probability of success is less than 0.3
 95 percent confidence interval:
 0.0000000 0.3206028
 sample estimates:
 probability of success
 0.24

```
confint(binom.test(33, 100, p = 1/3, conf.level = 0.9))
```

probability of success	lower	upper
0.3300000	0.2523035	0.4154543
level		
0.9000000		

```
binom.test(25, 100, p = 0.3, alt = "less", conf.level = 0.9)
```

Exact binomial test (with Score CI)

data: 25 out of 100
 number of successes = 25, number of trials = 100, p-value = 0.1631
 alternative hypothesis: true probability of success is less than 0.3
 90 percent confidence interval:
 0.0000000 0.3140311
 sample estimates:
 probability of success
 0.25

```
confint(binom.test(167, 500, p = 1/3))
```

probability of success	lower	upper
0.3340000	0.2927472	0.3772297
level		
0.9500000		

```
binom.test(146, 500, p = 0.3, alt = "less")
```

Exact binomial test (with Score CI)

data: 146 out of 500
 number of successes = 146, number of trials = 500, p-value = 0.3685
 alternative hypothesis: true probability of success is less than 0.3
 95 percent confidence interval:
 0.0000000 0.3273078
 sample estimates:
 probability of success
 0.292

```
confint(binom.test(33, 100, p = 1/3))
```

probability of success	lower	upper
0.3300000	0.2391985	0.4311728
level		
0.9500000		

```
binom.test(24, 100, p = 0.2, alt = "less")
```

Exact binomial test (with Score CI)

data: 24 out of 100

number of successes = 24, number of trials = 100, p-value = 0.8686

alternative hypothesis: true probability of success is less than 0.2

95 percent confidence interval:

0.0000000 0.3206028

sample estimates:

probability of success

0.24

4

Causation: Can We Say What Caused the Effect?

4.1 Association and Confounding

Example 4.1: Night Lights and Near-Sightedness

Often, when a dataset has only categorical variables, it may come in the form of a table and not a frame.

Here is a way to create a data frame in R.

NightLight1

	Darkness	NightLight	RoomLight
Near	18	78	41
Not	154	154	34

```
NightLight <- rbind(
  do(18) * data.frame(light = "Darkness", nearsight = "Near"),
  do(154) * data.frame(light = "Darkness", nearsight = "Not"),
  do(78) * data.frame(light = "NightLight", nearsight = "Near"),
  do(154) * data.frame(light = "NightLight", nearsight = "Not"),
  do(41) * data.frame(light = "RoomLight", nearsight = "Near"),
  do(34) * data.frame(light = "RoomLight", nearsight = "Not")
)
```

head(NightLight)

	light	nearsight	.row	.index
1	Darkness	Near	1	1
2	Darkness	Near	1	2
3	Darkness	Near	1	3
4	Darkness	Near	1	4
5	Darkness	Near	1	5
6	Darkness	Near	1	6

```
tally(nearsight ~ light, data = NightLight)
```

Table4.1

```

      light
nearsight Darkness NightLight RoomLight
Near          18          78          41
Not          154          154          34

```

```
tally(~nearsight | light, data = NightLight)
```

```

      light
nearsight Darkness NightLight RoomLight
Near          18          78          41
Not          154          154          34

```

```
tally(~nearsight + light, data = NightLight, margins = TRUE)
```

```

      light
nearsight Darkness NightLight RoomLight Total
Near          18          78          41    137
Not          154          154          34    342
Total        172          232          75    479

```

4.2 Observational studies versus experiments

Exploration 4.2: Have a Nice Trip

```
sim <- do(2) * rflip(12, 16/24)
sim
```

```

  n heads tails      prop
1 12     9     3 0.7500000
2 12     8     4 0.6666667

```

5

Comparing Two Proportions

5.1 Comparing Two Groups: Categorical Response

Example 5.1: Good and Bad Perceptions

```
head(GoodandBad, 30)
```

Table5.1

	wording	perception
1	goodyear	positive
2	goodyear	negative
3	badyear	positive
4	goodyear	positive
5	goodyear	negative
6	badyear	positive
7	goodyear	positive
8	goodyear	positive
9	goodyear	positive
10	badyear	negative
11	goodyear	negative
12	badyear	negative
13	goodyear	positive
14	badyear	negative
15	goodyear	positive
16	goodyear	positive
17	badyear	positive
18	goodyear	positive
19	goodyear	positive
20	goodyear	positive
21	badyear	negative
22	goodyear	positive
23	badyear	negative
24	goodyear	positive
25	badyear	negative
26	goodyear	positive
27	badyear	negative
28	goodyear	positive
29	badyear	positive
30	badyear	negative

Table 5.2

```
tally(perception ~ wording, data = GoodandBad, margins = TRUE)
```

	wording	
perception	badyear	goodyear
negative	8	3
positive	4	15
Total	12	18

```
tally(perception ~ wording, data = GoodandBad, format = "prop")
```

	wording	
perception	badyear	goodyear
negative	0.6666667	0.1666667
positive	0.3333333	0.8333333

```
prop(perception ~ wording, data = GoodandBad)
```

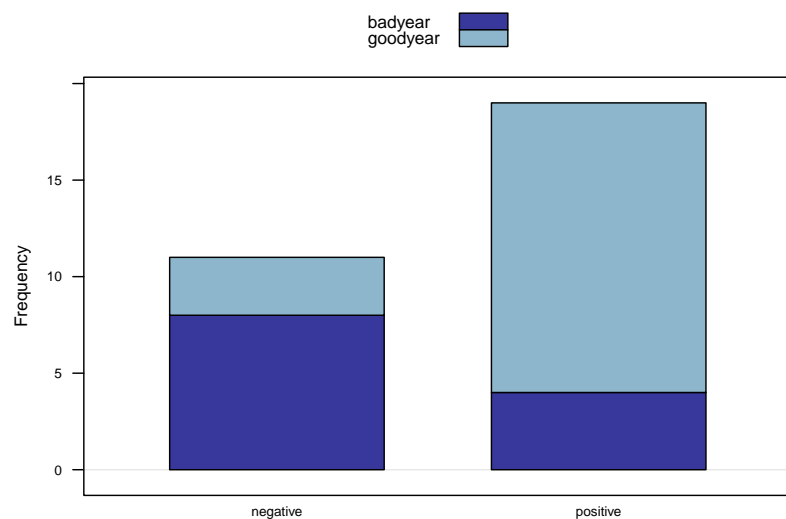
negative.badyear	negative.goodyear
0.6666667	0.1666667

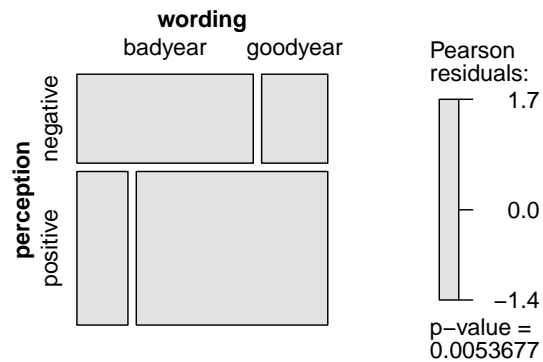
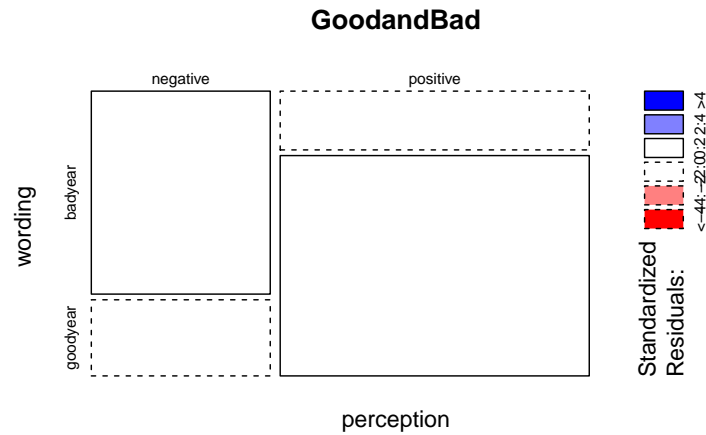
```
prop(perception ~ wording, level = "positive", data = GoodandBad)
```

positive.badyear	positive.goodyear
0.3333333	0.8333333

Figure 5.1

```
bargraph(~perception, groups = wording, data = GoodandBad, stack = TRUE, auto.key = TRUE)
mosaicplot(~perception + wording, data = GoodandBad, shade = TRUE)
mosaic(~perception + wording, data = GoodandBad, shade = TRUE)
```





Summarizing the data

Exploration 5.1: Murderous Nurse?

```
Nurse <- rbind(
  do(40) * data.frame(patient = "Death", shift = "Gilbert"),
  do(34) * data.frame(patient = "Death", shift = "NoGilbert"),
  do(217) * data.frame(patient = "NoDeath", shift = "Gilbert"),
  do(1350) * data.frame(patient = "NoDeath", shift = "NoGilbert")
)
```

Exploration5.1.7

```
tally(patient ~ shift, data = Nurse, margins = TRUE)
```

shift

Exploration5.1.7b

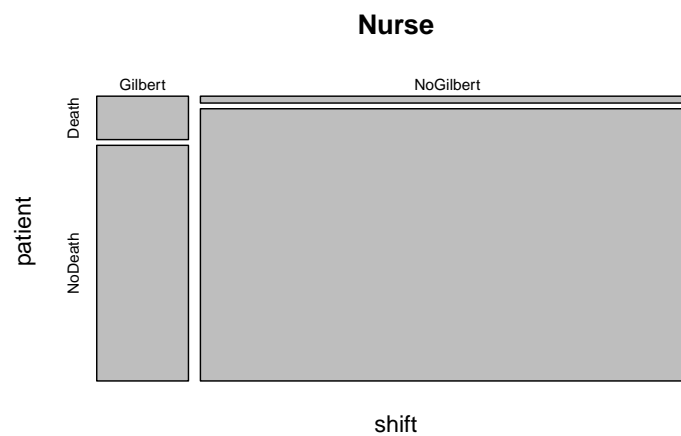
patient	Gilbert	NoGilbert
Death	40	34
NoDeath	217	1350
Total	257	1384

```
tally(patient ~ shift, data = Nurse, format = "prop") # conditional prop
```

	shift	
patient	Gilbert	NoGilbert
Death	0.15564202	0.02456647
NoDeath	0.84435798	0.97543353

```
mosaicplot(shift ~ patient, data = Nurse)
```

Exploration5.1.10



```
prop(patient ~ shift, data = Nurse)
```

Exploration5.1.14

Death.Gilbert	Death.NoGilbert
0.15564202	0.02456647

```
diffprop(patient ~ shift, data = Nurse)
```

```
diffprop  
-0.1310755
```

Further Analysis

```
Nurse2 <- rbind(
```

Exploration5.1.18

```
do(100) * data.frame(patient = "Death", shift = "Gilbert"),
do(357) * data.frame(patient = "Death", shift = "NoGilbert"),
do(157) * data.frame(patient = "NoDeath", shift = "Gilbert"),
do(1027) * data.frame(patient = "NoDeath", shift = "NoGilbert")
)
```

```
tally(patient ~ shift, data = Nurse2, margin = TRUE)
```

Exploration 5.1.18b

	shift	
patient	Gilbert	NoGilbert
Death	100	357
NoDeath	157	1027
Total	257	1384

```
tally(patient ~ shift, data = Nurse2, format = "prop")
```

	shift	
patient	Gilbert	NoGilbert
Death	0.3891051	0.2579480
NoDeath	0.6108949	0.7420520

```
diffprop(patient ~ shift, data = Nurse2) # diff in conditional prop
```

```
diffprop
-0.1311571
```

5.2 Comparing Two Properties: Simulation-Based Approach

Example 5.2: Swimming with Dolphins

```
head(Dolphin)
```

Table 5.3

```
swimming response
1 Dolphin Improve
2 Dolphin Improve
3 Dolphin Improve
4 Dolphin Improve
5 Dolphin Improve
6 Dolphin Improve
```

```
tally(response ~ swimming, data = Dolphin, margin = TRUE)
```

	swimming	
response	Control	Dolphin
Improve	3	10
NotImprove	12	5
Total	15	15

```
tally(response ~ swimming, data = Dolphin, format = "prop")
```

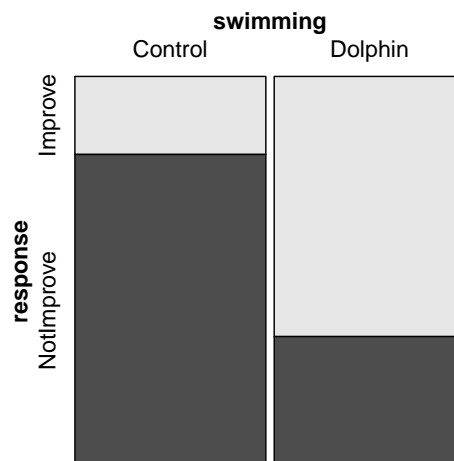
	swimming	
response	Control	Dolphin
Improve	0.2000000	0.6666667
NotImprove	0.8000000	0.3333333

```
diffprop(response ~ swimming, data = Dolphin)
```

```
diffprop
0.4666667
```

```
mosaic(response ~ swimming, data = Dolphin, dir = "v")
```

Figure5.2



```
mosaic(shuffle(response) ~ swimming, data = Dolphin, dir = "v")
```

Figure5.4

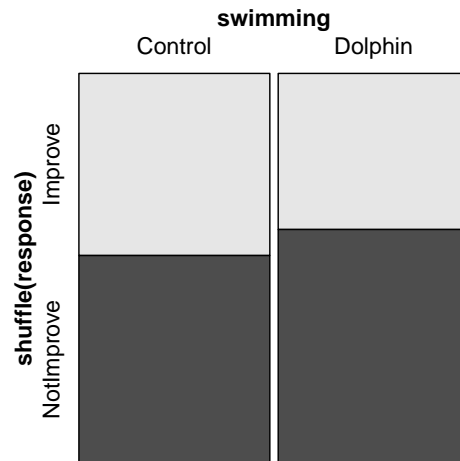


Figure 5.5

```
tally(shuffle(response) ~ swimming, data = Dolphin, margins = TRUE)
```

```
      swimming
shuffle(response) Control Dolphin
Improve           6         7
NotImprove        9         8
Total            15        15
```

```
tally(shuffle(response) ~ swimming, data = Dolphin, margins = TRUE)
```

```
      swimming
shuffle(response) Control Dolphin
Improve           4         9
NotImprove       11         6
Total            15        15
```

```
tally(shuffle(response) ~ swimming, data = Dolphin, margins = TRUE)
```

```
      swimming
shuffle(response) Control Dolphin
Improve           6         7
NotImprove        9         8
Total            15        15
```

```
diffprop(response ~ swimming, data = Dolphin)
```

```
diffprop
0.4666667
```

```
diffprop(shuffle(response) ~ swimming, data = Dolphin)
```

```
diffprop
0.06666667
```


[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

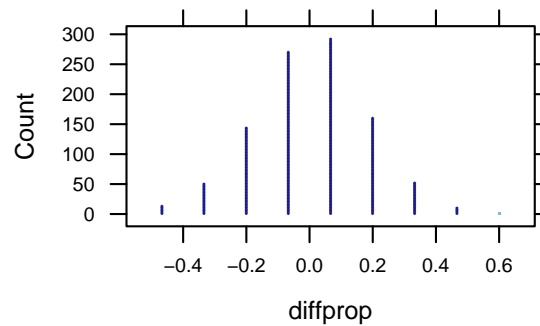
[illegible]

[illegible]

[illegible]

[illegible]

```
head(sim.dol, 3)
```



3. Strength of evidence:

```
favstats(~diffprop, data = sim.dol)
```

Figure5.6b

	min	Q1	median	Q3	max	mean	sd	n	missing
	-0.4666667	-0.0666667	0.0666667	0.0666667	0.6	0.004533333	0.1792156	1000	0

```
prop(~(diffprop >= 0.4667), data = sim.dol)
```

```
TRUE
0.001
```

Approximate test for difference in proportions:

```
prop.test(response ~ swimming, data = Dolphin)
```

Figure5.6c

2-sample test for equality of proportions with continuity correction

```
data:  tally(response ~ swimming)
X-squared = 4.8869, df = 1, p-value = 0.02706
alternative hypothesis: two.sided
95 percent confidence interval:
 -0.84620082 -0.08713252
sample estimates:
 prop 1    prop 2 
0.2000000 0.6666667
```

Estimation

Determining a 95% confidence interval using the 2SD Method and simulated standard deviation of the null distribution:

```
# given difference in sample proportions
diff <- diffprop(response ~ swimming, data = Dolphin)
# simulated standard deviation
sd <- sd(~diffprop, data = sim.dol)
# margin of error for 95% CI
```

Example5.2

```

MoE <- 2 * sd
MoE

[1] 0.3584312

# lower limit of 95% CI
diff - MoE

diffprop
0.1082355

# upper limit of 95% CI
diff + MoE

diffprop
0.8250979

```

Determining a 95% confidence interval using the approximate test for proportions:

```
confint(prop.test(response ~ swimming, data = Dolphin))
```

Example5.2b

prop 1	prop 2	lower	upper	level
0.20000000	0.66666667	-0.84620082	-0.08713252	0.95000000

Follow-up Analysis

```

Dolphin2 <- rbind(
  do(8) * data.frame(response = "Improve", swimming = "Control"),
  do(5) * data.frame(response = "Improve", swimming = "Dolphin"),
  do(7) * data.frame(response = "NotImprove", swimming = "Control"),
  do(10) * data.frame(response = "NotImprove", swimming = "Dolphin")
)

```

Figure5.7

```
tally(response ~ swimming, data = Dolphin2, margin = TRUE, format = "prop")
```

Figure5.7b

	swimming	
response	Control	Dolphin
Improve	0.5333333	0.3333333
NotImprove	0.4666667	0.6666667
Total	1.0000000	1.0000000

```
diffprop(response ~ swimming, data = Dolphin2)
```

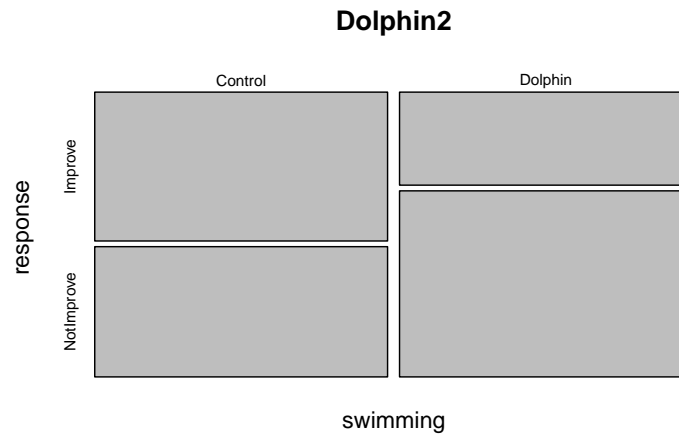
```

diffprop
-0.2

```

```
mosaicplot(swimming ~ response, data = Dolphin2)
```

Figure 5.7c



1. $H_0: \pi_{dolphins} - \pi_{control} = 0$
 $H_a: \pi_{dolphins} - \pi_{control} > 0$
 Test statistic: $\hat{p}_{dolphins} - \hat{p}_{control} = 0.20$ (the difference in the conditional sample proportions)
2. We simulate a world in which $\pi_{dolphins} - \pi_{control} = 0$:

```
sim.dol2 <- do(1000) * diffprop(shuffle(response) ~ swimming, data = Dolphin2)
```

Figure 5.7d

```
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
```

[illegible]

[illegible]

©2014

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

©2014

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

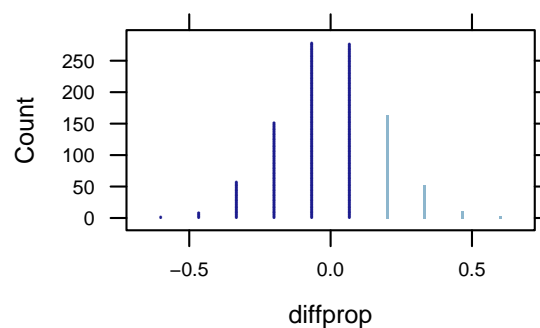
[illegible]

[illegible]

```
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
target level: Improve; other levels: NotImprove
```

```
head(sim.dol2, 3)
```

```
dotPlot(~diffprop, data = sim.dol2, groups = (diffprop >= 0.20),
        width = 1/15, cex = 5)
```



3. Strength of evidence:

```
favstats(~diffprop, data = sim.dol2)
```

Figure 5.7e

```
min      Q1 median      Q3 max      mean      sd      n missing
-0.6 -0.06666667      0 0.06666667 0.6 -0.0009333333 0.1796672 1000      0
```

```
prop(~(diffprop >= 0.2), data = sim.dol2)
```

```
TRUE
0.223
```

Approximate test for difference in proportions:

```
prop.test(response ~ swimming, data = Dolphin2, alt = "greater")
```

Figure5.7f

2-sample test for equality of proportions with continuity correction

```
data:  tally(response ~ swimming)
X-squared = 0.54299, df = 1, p-value = 0.2306
alternative hypothesis: greater
95 percent confidence interval:
 -0.1581698  1.0000000
sample estimates:
   prop 1    prop 2 
0.5333333 0.3333333
```

Relative Risk

Exploration 5.2: Is Yawning Contagious?

```
head(Yawning, 3)
```

Exploration5.2.9

```
  yawnSeed response
1  Seeded    Yawn
2  Seeded    Yawn
3  Seeded    Yawn
```

```
tally(response ~ yawnSeed, data = Yawning, margin = TRUE)
```

```
      yawnSeed
response Control Seeded
NoYawn      13     23
Yawn         3     11
Total       16     34
```

```
tally(response ~ yawnSeed, data = Yawning, format = "prop")
```

Exploration5.2.10

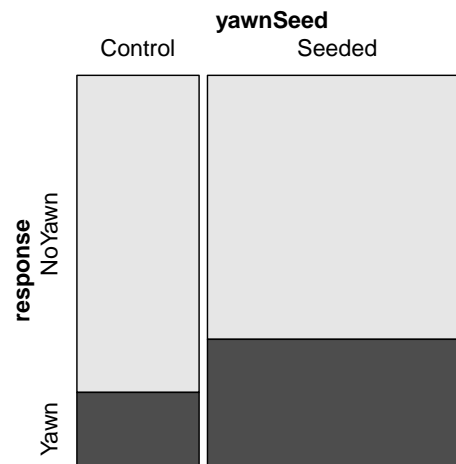
```
      yawnSeed
response Control Seeded
NoYawn 0.8125000 0.6764706
Yawn   0.1875000 0.3235294
```

```
diffprop(response ~ yawnSeed, level = "Yawn", data = Yawning)
```

```
diffprop
0.1360294
```

```
mosaic(response ~ yawnSeed, data = Yawning, dir = "v")
```

Exploration5.2.11



```
tally(shuffle(response) ~ yawnSeed, data = Yawning, margins = TRUE)
```

Exploration5.2.14

	yawnSeed	
shuffle(response)	Control	Seeded
NoYawn	9	27
Yawn	7	7
Total	16	34

1. $H_0: \pi_{seeded} - \pi_{control} = 0$

$H_a: \pi_{seeded} - \pi_{control} > 0$

Test statistic: $\hat{p}_{seeded} - \hat{p}_{control} = 0.136$ (the difference in the conditional sample proportions)

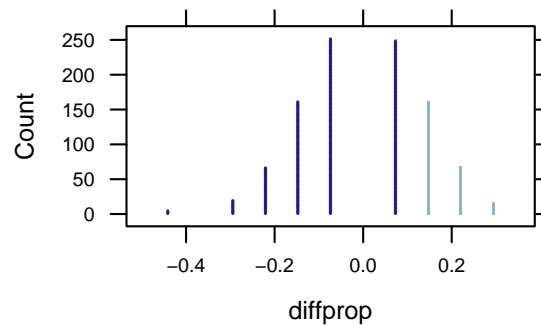
2. We simulate a world in which $\pi_{seeded} - \pi_{control} = 0$:

```
sim.yawn <-
  do(1000) * diffprop(shuffle(response) ~ yawnSeed, level = "Yawn", data = Yawning)
head(sim.yawn, 3)

      diffprop
1 -0.2316176
2  0.1360294
3  0.2279412

dotPlot(~diffprop, data = sim.yawn, groups = (diffprop >= 0.136), cex = 5)
```

Exploration5.2.16



3. Strength of evidence:

```
favstats(~diffprop, data = sim.yawn)
```

Exploration5.2.16b

	min	Q1	median	Q3	max	mean	sd	n
	-0.4154412	-0.1397059	-0.04779412	0.04411765	0.3198529	-0.005238971	0.1375262	1000
missing	0							

```
prop(~(diffprop >= 0.136), data = sim.yawn)
```

```
TRUE
0.245
```

Approximate test for difference in proportions:

```
prop.test(response ~ yawnSeed, data = Yawning, alt = "greater")
```

Exploration5.2.16c

Warning in stats::prop.test(t(table.from.formula), p = p, conf.level = conf.level, : Chi-squared approximation may be incorrect

2-sample test for equality of proportions with continuity correction

```
data:  tally(response ~ yawnSeed)
X-squared = 0.43786, df = 1, p-value = 0.2541
alternative hypothesis: greater
95 percent confidence interval:
 -0.1177157  1.0000000
sample estimates:
 prop 1    prop 2 
0.8125000 0.6764706
```

```
Yawning2 <- rbind(
  do(12) * data.frame(response = "NoYawn", yawnSeed = "Control"),
  do(24) * data.frame(response = "NoYawn", yawnSeed = "Seeded"),
  do(4) * data.frame(response = "Yawn", yawnSeed = "Control"),
  do(10) * data.frame(response = "Yawn", yawnSeed = "Seeded")
)
```

Exploration5.2.21

```
head(Yawning2, 3)
```

Exploration5.2.21b

```
  response yawnSeed .row .index
1   NoYawn   Control    1     1
2   NoYawn   Control    1     2
3   NoYawn   Control    1     3
```

```
tally(response ~ yawnSeed, data = Yawning2, margin = TRUE)
```

```
      yawnSeed
response Control Seeded
NoYawn      12     24
Yawn         4     10
Total       16     34
```

```
tally(response ~ yawnSeed, data = Yawning2, format = "prop")
```

Exploration5.2.21c

```
      yawnSeed
response Control Seeded
NoYawn 0.7500000 0.7058824
Yawn    0.2500000 0.2941176
```

```
diffprop(response ~ yawnSeed, level = "Yawn", data = Yawning2)
```

```
diffprop
0.04411765
```

$$1. H_0: \pi_{seeded} - \pi_{control} = 0$$

$$H_a: \pi_{seeded} - \pi_{control} > 0$$

Test statistic: $\hat{p}_{seeded} - \hat{p}_{control} = 0.0441$ (the difference in the conditional sample proportions)

2. We simulate a world in which $\pi_{seeded} - \pi_{control} = 0$:

```
sim.yawn2 <-
```

```
do(1000) * diffprop(shuffle(response) ~ yawnSeed, level = "Yawn", data = Yawning2)
```

Exploration5.2.23

```
target level: Yawn; other levels: NoYawn
target level: Yawn; other levels: NoYawn
target level: Yawn; other levels: NoYawn
target level: Yawn; other levels: NoYawn
target level: Yawn; other levels: NoYawn
target level: Yawn; other levels: NoYawn
target level: Yawn; other levels: NoYawn
target level: Yawn; other levels: NoYawn
target level: Yawn; other levels: NoYawn
target level: Yawn; other levels: NoYawn
```

[illegible]

©2014

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

©2014

[illegible]

[illegible]

[illegible]

©2014

[illegible]

[illegible]

[illegible]

[illegible]

diffprop bin (approx.)	Count
-0.35 to -0.30	5
-0.30 to -0.25	15
-0.25 to -0.20	80
-0.20 to -0.15	155
-0.15 to -0.10	225
-0.10 to -0.05	230
-0.05 to 0.00	250
0.00 to 0.05	165
0.05 to 0.10	90
0.10 to 0.15	10
0.15 to 0.20	5

Last Modified: May 19, 2015

```
favstats(~diffprop, data = sim.yawn2)
```

Exploration5.2.23b

```
      min      Q1    median      Q3      max      mean      sd    n missing
-0.4154412 -0.1397059 0.04411765 0.1360294 0.4117647 -0.001102941 0.1417443 1000      0
```

```
prop(~(diffprop >= 0.0441), data = sim.yawn2)
```

```
TRUE
0.518
```

Approximate test for difference in proportions:

```
prop.test(response ~ yawnSeed, data = Yawning2, alt = "greater")
```

Exploration5.2.23c

2-sample test for equality of proportions with continuity correction

```
data:  tally(response ~ yawnSeed)
X-squared = 3.853e-31, df = 1, p-value = 0.5
alternative hypothesis: greater
95 percent confidence interval:
 -0.2196049  1.0000000
sample estimates:
 prop 1    prop 2 
0.7500000 0.7058824
```

Estimation

```
sd <- sd(~diffprop, data = sim.yawn2)
sd
```

Exploration5.2.24

```
[1] 0.1417443
```

Determining a 95% confidence interval using the 2SD Method and simulated standard deviation of the null distribution:

```
# given difference in sample proportions
diff <- diffprop(response ~ yawnSeed, level = "Yawn", data = Yawning2)
# previously found simulated standard deviation
sd
```

Exploration5.2.24b

```
[1] 0.1417443
```

```
# margin of error for 95% CI
MoE <- 2 * sd
MoE
```

```
[1] 0.2834887
```

```
# lower limit of 95% CI
diff - MoE

diffprop
-0.239371

# upper limit of 95% CI
diff + MoE

diffprop
0.3276063
```

Determining a 95% confidence interval using the approximate test for proportions:

```
confint(prop.test(response ~ yawnSeed, data = Yawning2))
```

Exploration5.2.24c

```
      prop 1      prop 2      lower      upper      level
0.7500000  0.7058824 -0.2616754  0.3499107  0.9500000
```

Effect of Sample Size

```
Yawning3 <- rbind(
  do(240) * data.frame(response = "NoYawn", yawnSeed = "Control"),
  do(120) * data.frame(response = "NoYawn", yawnSeed = "Seeded"),
  do(100) * data.frame(response = "Yawn", yawnSeed = "Control"),
  do(40)  * data.frame(response = "Yawn", yawnSeed = "Seeded")
)
```

Exploration5.2.31

```
head(Yawning3, 3)
```

Exploration5.2.31b

```
  response yawnSeed .row .index
1  NoYawn  Control   1     1
2  NoYawn  Control   1     2
3  NoYawn  Control   1     3
```

```
tally(response ~ yawnSeed, data = Yawning3, margin = TRUE)
```

```
      yawnSeed
response Control Seeded
NoYawn      240    120
Yawn        100     40
Total       340    160
```

$$1. H_0: \pi_{seeded} - \pi_{control} = 0$$

$$H_a: \pi_{seeded} - \pi_{control} > 0$$

Test statistic: $\hat{p}_{seeded} - \hat{p}_{control} = 0.0441$ (the difference in the conditional sample proportions)

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

©2014

[illegible]

©2014

[illegible]

[illegible]

[illegible]

©2014

[illegible]

©2014

[illegible]

©2014

[illegible]

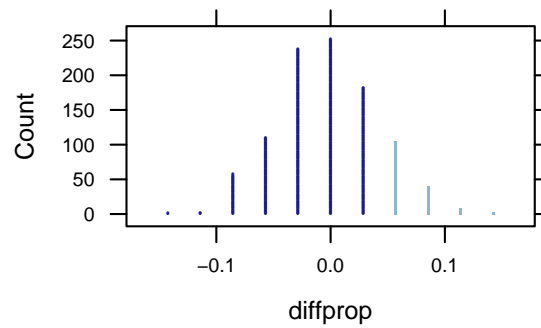
[illegible]

[illegible]

[illegible]

[illegible]

```
dotPlot(~diffprop, data = sim.yawn3, groups = (diffprop >= 0.0441), cex = 5)
```



3. Strength of evidence:

```
favstats(~diffprop, data = sim.yawn3)
```

Exploration5.2.32b

	min	Q1	median	Q3	max	mean	sd	n
	-0.1452206	-0.02573529	0.001838235	0.02941176	0.1397059	-0.001378676	0.04257306	1000
missing								
0								

```
prop(~(diffprop >= 0.0441), data = sim.yawn3)
```

```
TRUE
0.151
```

Approximate test for difference in proportions:

```
prop.test(response ~ yawnSeed, data = Yawning3, alt = "greater")
```

Exploration5.2.32c

2-sample test for equality of proportions with continuity correction

```
data: tally(response ~ yawnSeed)
X-squared = 0.84298, df = 1, p-value = 0.8207
alternative hypothesis: greater
95 percent confidence interval:
-0.1181584 1.0000000
sample estimates:
prop 1    prop 2
0.7058824 0.7500000
```

Relative risk

5.3 Comparing Two Proportions: Theory-Based Approach

Example 5.3: Smoking and Birth Gender

```
head(Smoking, 3)
```

Figure5.9

```
  parents child
1 smokers  girl
2 smokers  girl
3 smokers  girl
```

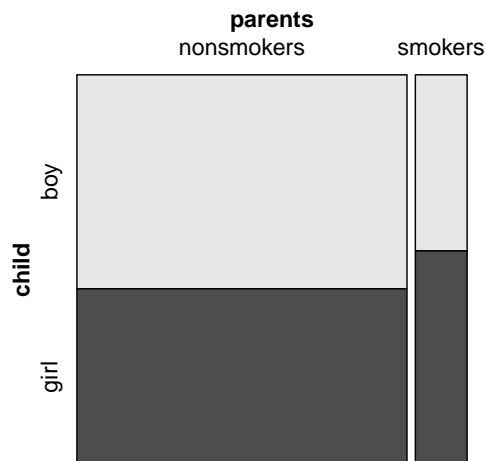
```
summary(Smoking)
```

```
  parents      child
Length:4167  Length:4167
Class :character Class :character
Mode  :character  Mode  :character
```

```
tally(parents ~ child, data = Smoking, margin = TRUE)
```

```
      child
parents boy girl
nonsmokers 1975 1627
smokers    255  310
Total    2230 1937
```

```
mosaic(child ~ parents, data = Smoking, dir = "v")
```



```
tally(child ~ parents, data = Smoking, format = "prop", margins = TRUE)
```

Figure5.10

```
      parents
child nonsmokers smokers
boy    0.5483065 0.4513274
girl    0.4516935 0.5486726
Total  1.0000000 1.0000000
```

```
diffprop(child ~ parents, data = Smoking)
```

```
diffprop
-0.09697906
```

1. $H_0: \pi_{\text{smoker}} - \pi_{\text{nonsmoker}} = 0$

$H_a: \pi_{\text{smoker}} - \pi_{\text{nonsmoker}} \neq 0$

Test statistic: $\hat{p}_{\text{smoker}} - \hat{p}_{\text{nonsmoker}} = -0.097$ (the difference in the conditional sample proportions)

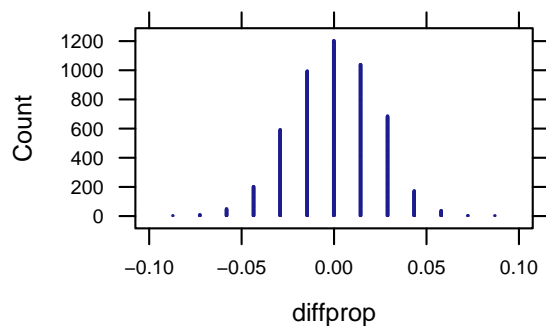
2. We simulate a world in which $\pi_{\text{smoker}} - \pi_{\text{nonsmoker}} = 0$:

```
sim.smoke <- do(5000) * diffprop(shuffle(child) ~ parents, data = Smoking)
head(sim.smoke, 3)
```

Figure5.10b

```
diffprop
1 0.023825505
2 -0.002792451
3 0.027920575
```

```
dotPlot(~diffprop, data = sim.smoke, cex = 25)
```



3. Strength of evidence:

```
favstats(~diffprop, data = sim.smoke)
```

Figure5.10c

```
      min      Q1      median      Q3      max      mean      sd      n
-0.08264632 -0.01507766 -0.0007449156 0.01563536 0.09139416 -0.0003726737 0.0225623 5000
missing
0
```

```
prop(~(diffprop <= -0.097 | diffprop >= 0.097), data = sim.smoke)
```

```
TRUE
0
```

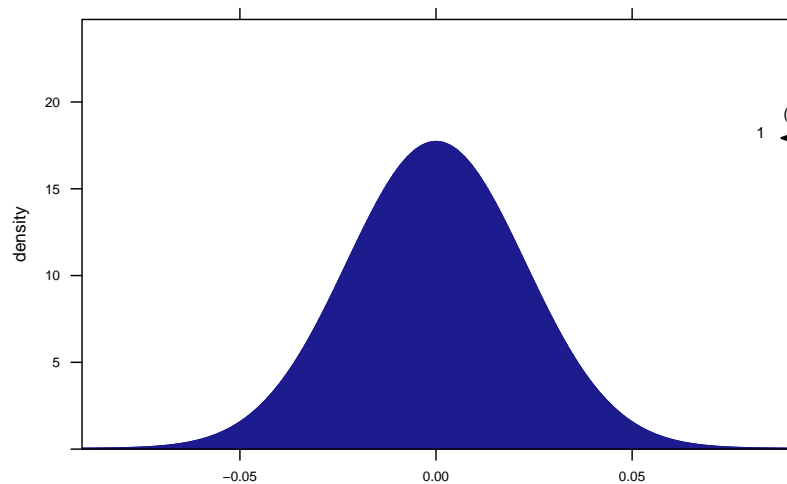
Normal approximation (using simulated standard deviation):

```
sd <- sd(~diffprop, data = sim.smoke)
2 * xpnorm(0.097, 0, sd, lower.tail = FALSE) # 2 times because two-sided
```

Figure5.11

If $X \sim N(0, 0.0225622971246741)$, then

```
P(X <= 0.097) = P(Z <= 4.299) = 1
P(X > 0.097) = P(Z > 4.299) = 0
[1] 1.714098e-05
```



Approximate test for difference in proportions:

```
prop.test(child ~ parents, data = Smoking)
```

Figure5.12

2-sample test for equality of proportions with continuity correction

```
data: tally(child ~ parents)
X-squared = 18.077, df = 1, p-value = 2.122e-05
alternative hypothesis: two.sided
95 percent confidence interval:
 0.05182158 0.14213654
sample estimates:
 prop 1    prop 2 
0.5483065 0.4513274
```

Estimation

```
confint(prop.test(child ~ parents, data = Smoking))
```

Figure5.13

```
      prop 1    prop 2    lower    upper    level
0.54830650 0.45132743 0.05182158 0.14213654 0.95000000
```

```
confint(prop.test(child ~ parents, data = Smoking, conf.level = 0.99))
```

Figure5.14

```
      prop 1      prop 2      lower      upper      level
0.54830650 0.45132743 0.03795376 0.15600436 0.99000000
```

Formulas

```
prop(child ~ parents, data = Smoking)
```

Example5.3

```
boy.nonsmokers    boy.smokers
      0.5483065      0.4513274
```

```
p.1 <- 0.548
p.2 <- 0.451
p.hat <- prop(~child, data = Smoking)
p.hat # pooled prop of success
```

```
      boy
0.5351572
```

```
n.1 <- 565
n.2 <- 3602
```

```
z <- (p.1 - p.2)/sqrt((p.hat * (1 - p.hat) * (1/n.1 + 1/n.2)))
z
```

Example5.3b

```
      boy
4.29796
```

```
SE <- sqrt(p.1 * (1 - p.1)/n.1 + p.2 * (1 - p.2)/n.2)
SE
```

Example5.3c

```
[1] 0.02251975
```

```
MoE <- 2 * SE
MoE
```

Example5.3d

```
[1] 0.04503951
```

Exploration 5.3: Donating Blood

```
sample(Blood, 5)
```

Exploration5.3.2

```
      year response orig.ids
206  2002  donated      206
482  2002 did.not      482
2327 2004 did.not     2327
1712 2004 did.not     1712
146   2002  donated      146
```

```
tally(response ~ year, data = Blood, format = "count", margin = TRUE)
```

```
      year
response 2002 2004
did.not  1152 1106
donated   210  230
Total    1362 1336
```

```
tally(response ~ year, data = Blood, format = "prop")
```

Exploration5.3.3

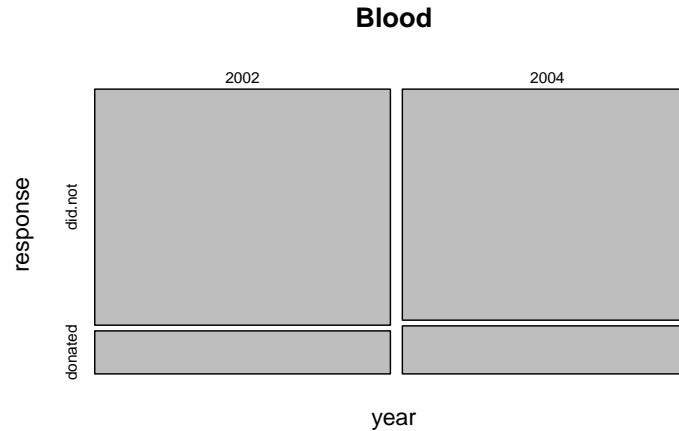
```
      year
response 2002 2004
did.not  0.8458150 0.8278443
donated  0.1541850 0.1721557
```

```
diffprop(response ~ year, level = "donated", data = Blood)
```

```
diffprop
0.01797067
```

```
mosaicplot(year ~ response, data = Blood)
```

Exploration5.3.4



1. $H_0: \pi_{2004} - \pi_{2002} = 0$
 $H_a: \pi_{2004} - \pi_{2002} \neq 0$
 Test statistic: $\hat{p}_{2004} - \hat{p}_{2002} = 0.0180$ (the difference in the conditional sample proportions)
2. We simulate a world in which $\pi_{2004} - \pi_{2002} = 0$:

```
sim.blood <-  
  do(5000) * diffprop(shuffle(response) ~ year, level = "donated", data = Blood)
```

Exploration 5.3.6

[illegible]

[illegible]

[illegible]

©2014

Last Modified: May 19, 2015

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

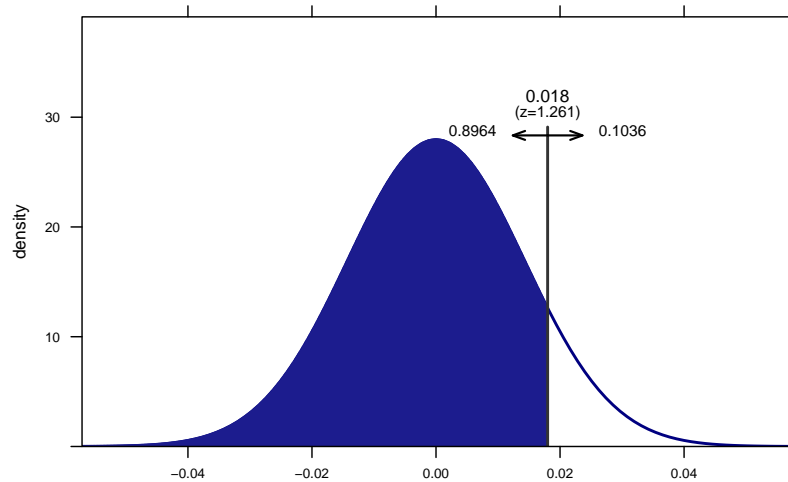
[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



Approximate test for difference in proportions:

```
prop.test(response ~ year, data = Blood)
```

Exploration5.3.11

2-sample test for equality of proportions with continuity correction

```
data:  tally(response ~ year)
X-squared = 1.4668, df = 1, p-value = 0.2258
alternative hypothesis: two.sided
95 percent confidence interval:
 -0.01065634  0.04659768
sample estimates:
   prop 1    prop 2 
0.8458150 0.8278443
```

```
confint(prop.test(response ~ year, data = Blood))
```

Exploration5.3.10

```
      prop 1    prop 2    lower    upper    level 
0.84581498 0.82784431 -0.01065634 0.04659768 0.95000000
```

```
Blood2 <- rbind(
  do(239) * data.frame(response = "donated", Sex = "Male"),
  do(201) * data.frame(response = "donated", Sex = "Female"),
  do(1032) * data.frame(response = "did.not", Sex = "Male"),
  do(1226) * data.frame(response = "did.not", Sex = "Female")
)
```

Exploration5.3.15

```
tally(response ~ Sex, data = Blood2, margin = TRUE)
```

Exploration5.3.15b

```

      Sex
response Male Female
donated   239    201
did.not  1032   1226
Total    1271   1427

```

```
tally(response ~ Sex, data = Blood2, format = "prop")
```

```

      Sex
response Male   Female
donated  0.1880409 0.1408549
did.not  0.8119591 0.8591451

```

```
diffprop(response ~ Sex, data = Blood2)
```

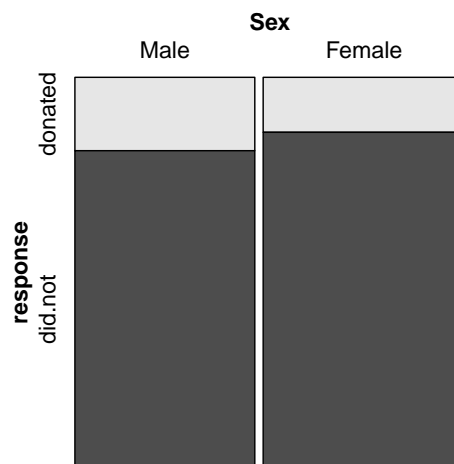
```

diffprop
-0.04718597

```

Exploration5.3.15c

```
mosaic(response ~ Sex, data = Blood2, dir = "v")
```



$$1. H_0: \pi_{female} - \pi_{male} = 0$$

$$H_a: \pi_{female} - \pi_{male} \neq 0$$

Test statistic: $\hat{p}_{female} - \hat{p}_{male} = -0.0472$ (the difference in the conditional sample proportions)

2. We simulate a world in which $\pi_{female} - \pi_{male} = 0$:

```
sim.blood2 <- do(5000) * diffprop(shuffle(response) ~ Sex, data = Blood2)
```

Exploration5.3.15d

```
target level: donated; other levels: did.not
```

```
target level: donated; other levels: did.not
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

Last Modified: May 19, 2015

©2014

[illegible]

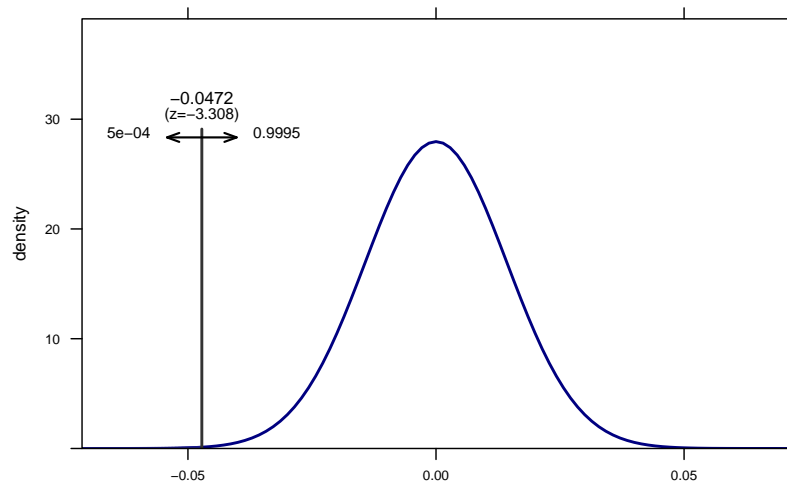
[illegible]

[illegible]

[illegible]

[illegible]

©2014



Approximate test for difference in proportions:

```
prop.test(response ~ Sex, data = Blood2)
```

Exploration5.3.15g

2-sample test for equality of proportions with continuity correction

```
data:  tally(response ~ Sex)
X-squared = 10.623, df = 1, p-value = 0.001117
alternative hypothesis: two.sided
95 percent confidence interval:
 0.01838452 0.07598742
sample estimates:
  prop 1    prop 2 
0.1880409 0.1408549
```

6

Comparing Two Means

6.1 Comparing Two Groups: Quantitative Response

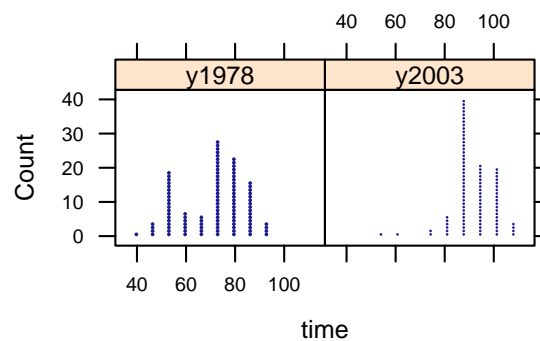
Example 6.1: Geyser Eruptions

```
head(OldFaithful, 3)
```

```
  year time
1 y1978  78
2 y1978  74
3 y1978  68
```

```
dotPlot(~time | year, data = OldFaithful)
```

Figure6.1



```
fivenum(~time, data = OldFaithful)
```

```
[1] 42 73 84 91 110
```

```
fivenum(time ~ year, data = OldFaithful)
```

Example6.1

```
y19781 y19782 y19783 y19784 y19785 y20031 y20032 y20033 y20034 y20035
42.0   59.0   75.0   80.5   95.0   56.0   87.0   91.0   97.0  110.0
```

```
IQR(~time, data = OldFaithful)
```

Example6.1b

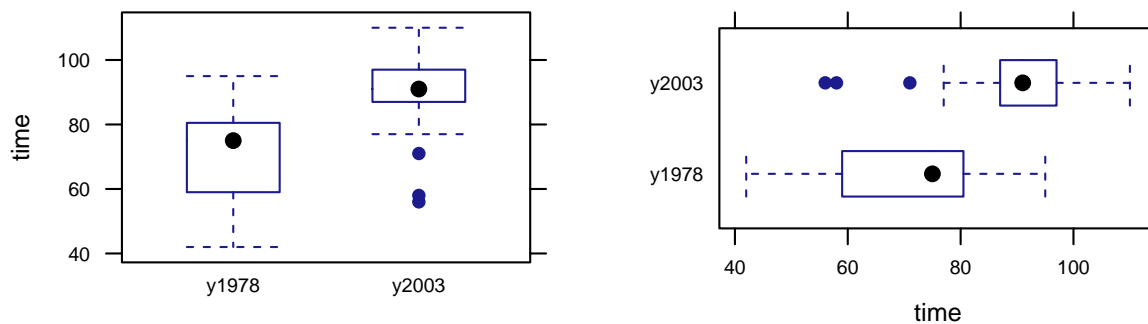
```
[1] 18
```

```
IQR(~time | year, data = OldFaithful)
```

```
y1978 y2003
20.75 10.00
```

```
bwplot(time ~ year, data = OldFaithful)
bwplot(year ~ time, data = OldFaithful, horizontal = TRUE)
```

Figure6.2



Exploration 6.1A: Haircut Prices

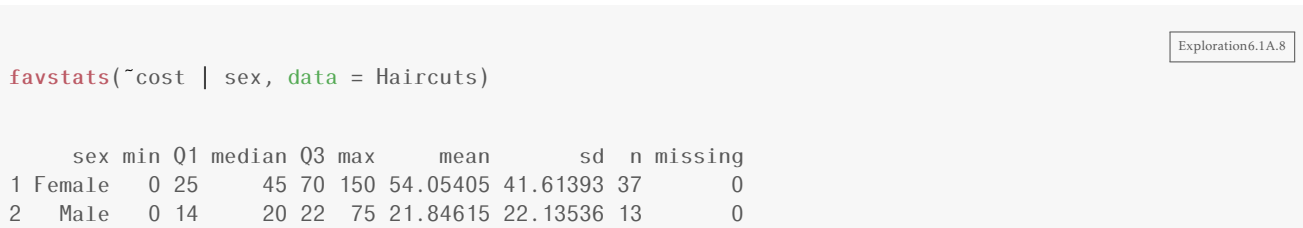
```
head(Haircuts)
```

Exploration6.1A.1

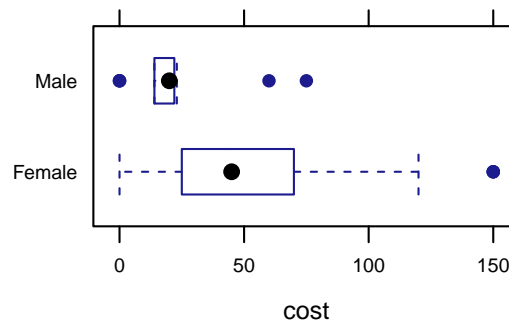
```
sex cost
1 Female 50
2 Male 20
3 Female 60
4 Male 75
5 Female 150
6 Male 23
```

```
dotPlot(~cost | sex, data = Haircuts, width = 1, cex = 0.25)
```

Exploration6.1A.4



Last Modified: May 19, 2015



Exploration6.1A.18

```
IQR(cost ~ sex, data = Haircuts)
```

Female	Male
45	8

6.2 Comparing Two Means: Simulation-Based Approach

Example 6.2: Bicycling to Work

```
head(BikeTimes)
```

Table6.2

	frame	time
1	steel	115.7500
2	steel	115.6667
3	steel	108.7333
4	steel	117.7333
5	steel	112.6167
6	steel	109.5667

Figure6.3

```
bwplot(frame ~ time, data = BikeTimes, horizontal = TRUE)
```

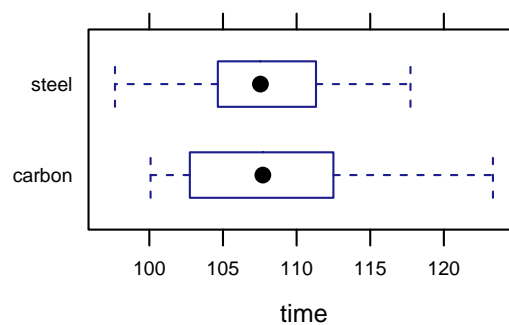


Table6.3

```
favstats(time ~ frame, data = BikeTimes)
```

	frame	min	Q1	median	Q3	max	mean	sd	n	missing
1	carbon	100.08333	102.7875	107.7083	112.4833	123.3333	108.3436	6.248036	26	0
2	steel	97.66667	104.6750	107.5417	111.2458	117.7333	107.8089	4.891712	30	0

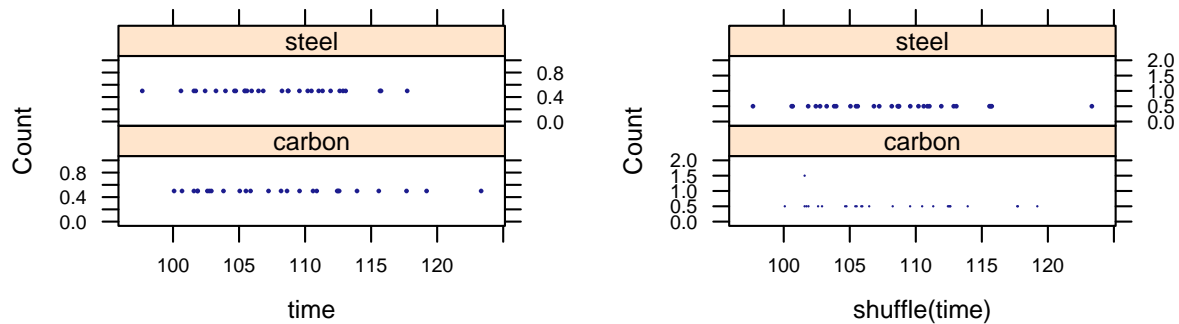
Figure6.4

```
dotPlot(~time | frame, data = BikeTimes, width = 0.01, cex = 0.1, layout = c(1, 2))
diffmean(time ~ frame, data = BikeTimes)
```

```
diffmean
-0.5347007
```

```
dotPlot(~shuffle(time) | frame, data = BikeTimes, width = 0.01, cex = 0.1, layout = c(1, 2))
diffmean(shuffle(time) ~ frame, data = BikeTimes)
```

```
diffmean
-0.9355555
```



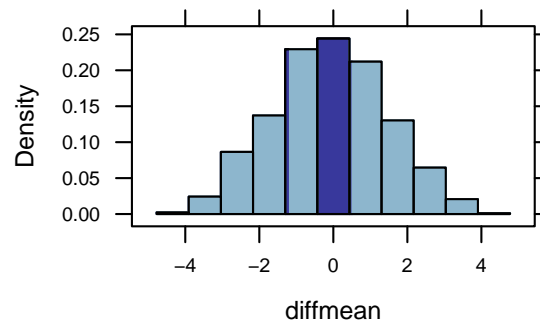
1. $H_0: \mu_{\text{carbon}} - \mu_{\text{steel}} = 0$
 $H_a: \mu_{\text{carbon}} - \mu_{\text{steel}} \neq 0$
 Test statistic: $\bar{x}_{\text{carbon}} - \bar{x}_{\text{steel}} = 0.53$ (the difference in the sample means)
2. We simulate a world in which $\mu_{\text{carbon}} - \mu_{\text{steel}} = 0$:

Figure6.7

```
sim.bike <- do(1000) * diffmean(shuffle(time) ~ frame, data = BikeTimes)
head(sim.bike, 3)
```

```
diffmean
1 -1.137777
2 2.1276922
3 -0.5442736
```

```
histogram(~ diffmean, data = sim.bike,
  groups = (diffmean <= -0.53 | diffmean >= 0.53))
```



3. Strength of evidence:

```
favstats(~diffmean, data = sim.bike)
```

Figure6.8

	min	Q1	median	Q3	max	mean	sd	n	missing
	-4.504957	-1.118932	-0.0985471	0.9619231	4.173846	-0.08281675	1.51832	1000	0

```
prop(~(diffmean <= -0.53 | diffmean >= 0.53), data = sim.bike)
```

```
TRUE
0.74
```

Estimating a confidence interval

Determining a 95% confidence interval using the 2SD Method and standard deviation of the null distribution:

```
diff <- -diffmean(time ~ frame, data = BikeTimes) # note the negative sign
sd <- sd(~diffmean, data = sim.bike)
diff - 2 * sd # lower limit of 95% CI
```

Example6.2

```
diffmean
-2.501939
```

```
diff + 2 * sd # upper limit of 95% CI
```

```
diffmean
3.57134
```

Exploration 6.2: Lingering Effects of Sleep Deprivation

```
head(Sleep)
```

Exploration6.2.2


```

      sleep time
1 unrestricted -7.0
2 unrestricted 11.6
3 unrestricted 12.1
4 unrestricted 12.6
5 unrestricted 14.5
6 unrestricted 18.6

```

Exploration6.2.5

```

dotPlot(~time | sleep, data = Sleep, cex = 0.5, width = 1, layout = c(1, 2))
favstats(time ~ sleep, data = Sleep)

```

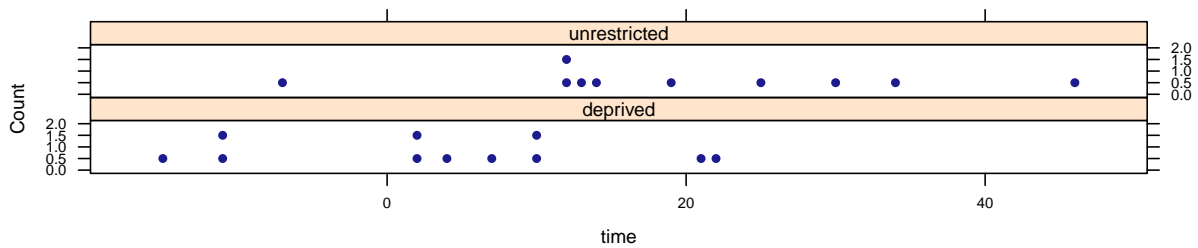
	sleep	min	Q1	median	Q3	max	mean	sd	n	missing
1	deprived	-14.7	-4.250	4.50	9.800	21.8	3.90	12.17185	11	0
2	unrestricted	-7.0	12.225	16.55	29.175	45.6	19.82	14.72532	10	0

```
diff(mean(time ~ sleep, data = Sleep))
```

```

unrestricted
15.92

```



Exploration6.2.9

```
diff(mean(shuffle(time) ~ sleep, data = Sleep))
```

```

unrestricted
3.95

```

```

sample <- do(30) * diff(mean(shuffle(time) ~ sleep, data = Sleep))
head(sample, 3)

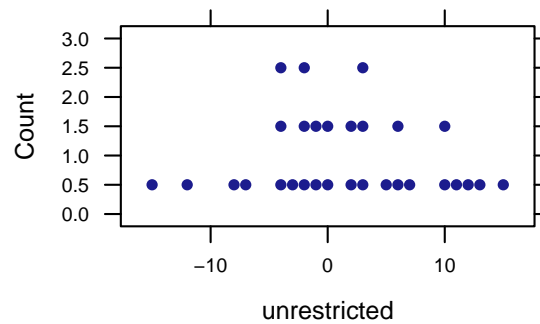
```

```

unrestricted
1 -0.5936364
2 10.7463636
3 14.9463636

```

```
dotPlot(~unrestricted, data = sample, width = 1, cex = 0.25)
```



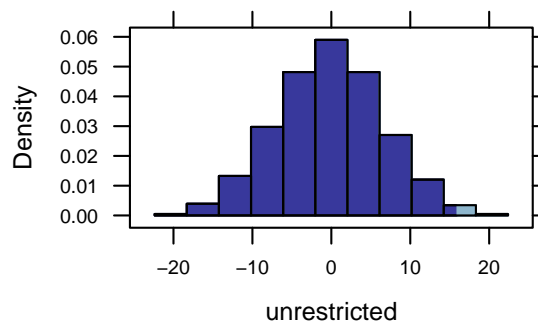
1. $H_0: \mu_{\text{unrestricted}} - \mu_{\text{deprived}} = 0$
 $H_a: \mu_{\text{unrestricted}} - \mu_{\text{deprived}} > 0$
 Test statistic: $\bar{x}_{\text{unrestricted}} - \bar{x}_{\text{deprived}} = 15.92$ (the difference in the sample means)
2. We simulate a world in which $\mu_{\text{unrestricted}} - \mu_{\text{deprived}} = 0$:

```
sim.sleep <- do(1000) * diff(mean(shuffle(time) ~ sleep, data = Sleep))
head(sim.sleep, 3)
```

	unrestricted
1	3.377273
2	8.570000
3	-4.755455

```
histogram(~ unrestricted, data = sim.sleep,
          groups = (unrestricted >= 15.92))
```

Exploration6.2.10



3. Strength of evidence:

```
favstats(~unrestricted, data = sim.sleep)
```

	min	Q1	median	Q3	max	mean	sd	n	missing
	-18.46273	-4.836591	-0.3263636	4.226818	22.23909	-0.2376673	6.774061	1000	0

```
prop(~(unrestricted >= 15.92), data = sim.sleep)
```

```
TRUE
0.01
```

Exploration6.2.10b

Determining a 95% confidence interval using the 2SD Method and standard deviation of the null distribution:

```
diff <- diff(mean(time ~ sleep, data = Sleep))
sd <- sd(~unrestricted, data = sim.sleep)
diff - 2 * sd # lower limit of 95% CI
```

Exploration6.2.13

```
unrestricted
2.371878
```

```
diff + 2 * sd # upper limit of 95% CI
```

```
unrestricted
29.46812
```

Another statistic

```
median(time ~ sleep, data = Sleep)
```

Exploration6.2.16

```
deprived unrestricted
4.50      16.55
```

```
diff(median(time ~ sleep, data = Sleep))
```

```
unrestricted
12.05
```

$$1. H_0: \text{median}_{\text{unrestricted}} - \text{median}_{\text{deprived}} = 0$$

$$H_a: \text{median}_{\text{unrestricted}} - \text{median}_{\text{deprived}} > 0$$

Test statistic: $\text{median}_{\text{unrestricted}} - \text{median}_{\text{deprived}} = 12.05$ (the difference in the sample medians)

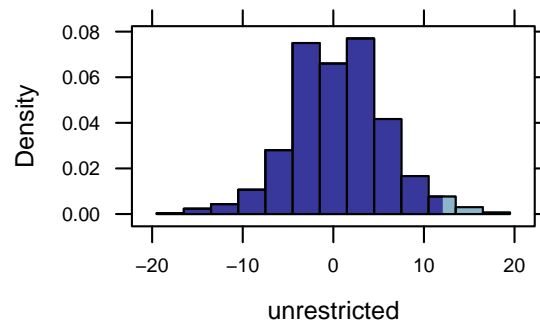
2. We simulate a world in which $\text{median}_{\text{unrestricted}} - \text{median}_{\text{deprived}} = 0$:

```
sim.med <- do(1000) * diff(median(shuffle(time) ~ sleep, data = Sleep))
head(sim.med, 3)
```

Exploration6.2.16b

```
unrestricted
1      3.05
2      3.95
3      2.10
```

```
histogram(~ unrestricted, data = sim.med,
  groups = (unrestricted >= 12.05),
  width = 3)
```



3. Strength of evidence:

```
favstats(~unrestricted, data = sim.med)

  min   Q1 median   Q3   max   mean    sd   n missing
-16.5 -2.3   -0.5  3.55 17.05 0.4765 5.19492 1000      0

prop(~(unrestricted >= 12.05), data = sim.med)

TRUE
0.021
```

Exploration6.2.16c

6.3 Comparing Two Means: Theory-Based Approach

Example 6.3: Breastfeeding and Intelligence

```
head(BreastFeedIntell)
```

Table6.4

```
  feeding   GCI
1 Breastfed 126.701
2 Breastfed 124.692
3 Breastfed  99.787
4 Breastfed 104.966
5 Breastfed  97.252
6 Breastfed 131.276
```

```
favstats(GCI ~ feeding, data = BreastFeedIntell)
```

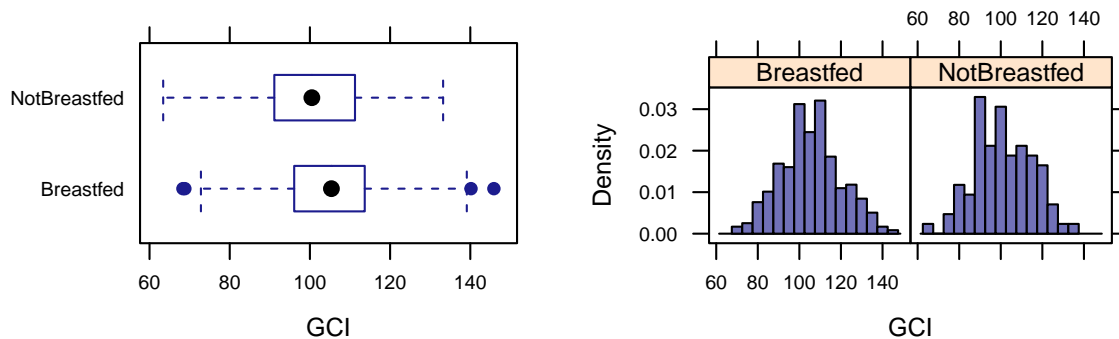
```
  feeding   min   Q1 median   Q3   max   mean    sd   n missing
1 Breastfed 68.330 96.083 105.366 113.677 145.889 105.3 14.49998 237      0
2 NotBreastfed 63.408 91.127 100.485 111.243 133.226 100.9 13.99997  85      0
```

```
diffmean(GCI ~ feeding, data = BreastFeedIntell)
```

```
diffmean
-4.40005
```

```
bwplot(feeding ~ GCI, horizontal = TRUE, data = BreastFeedIntell)
histogram(~GCI | feeding, data = BreastFeedIntell, width = 5)
```

Figure6.10



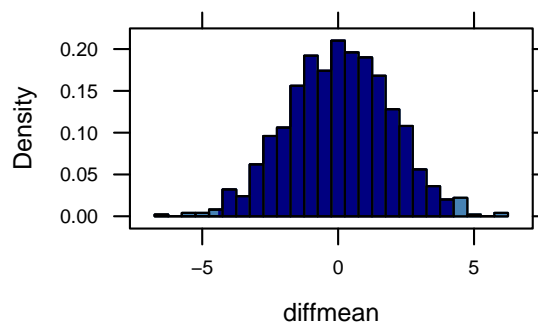
1. $H_0: \mu_{\text{breastfed}} - \mu_{\text{not}} = 0$
 $H_a: \mu_{\text{breastfed}} - \mu_{\text{not}} \neq 0$
 Test statistic: $\bar{x}_{\text{breastfed}} - \bar{x}_{\text{not}} = 4.40$ (the difference in the sample means)
2. We simulate a world in which $\mu_{\text{breastfed}} - \mu_{\text{not}} = 0$:

```
sim.GCI <- do(1000) * diffmean(shuffle(GCI) ~ feeding, data = BreastFeedIntell)
head(sim.GCI, 3)

diffmean
1 3.511704
2 -1.579765
3 -2.509369

histogram(~ diffmean, data = sim.GCI, width = 0.5,
  group = cut(diffmean, c(-7, -4.40, 4.40, 7)),
  fcol = c("steelblue", "navy", "steelblue"))
```

Figure6.11



3. Strength of evidence:

```
favstats(~diffmean, data = sim.GCI)
```

Figure6.12

```
      min      Q1      median      Q3      max      mean      sd      n missing
-6.398096 -1.225017 0.07139459 1.391159 6.117258 0.05488575 1.884651 1000      0
```

```
prop(~(diffmean <= -4.4 | diffmean >= 4.4), data = sim.GCI)
```

```
TRUE
0.021
```

Determining a 95% confidence interval using the 2SD Method and standard deviation of the null distribution:

```
diff <- -diffmean(GCI ~ feeding, data = BreastFeedIntell) # note the negative sign
sd <- sd(~diffmean, data = sim.GCI)
sd
```

Example6.3

```
[1] 1.884651
```

```
diff - 2 * sd # lower limit of 95% CI
```

```
diffmean
0.6307471
```

```
diff + 2 * sd # upper limit of 95% CI
```

```
diffmean
8.169352
```

```
t.test(GCI ~ feeding, data = BreastFeedIntell)
```

Figure6.13

```
Welch Two Sample t-test
```

```
data: GCI by feeding
t = 2.4624, df = 153.01, p-value = 0.01491
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.8698749 7.9302245
sample estimates:
```

```
mean in group Breastfed mean in group NotBreastfed
              105.3              100.9
```

```
stat(t.test(GCI ~ feeding, data = BreastFeedIntell))
```

```
t
2.462397
```

Exploration 6.3: Close Friends

```
head(CloseFriends)
```

Exploration6.3.1

```
sex friends
1 Men      0
2 Men      0
3 Men      0
4 Men      0
5 Men      0
6 Men      0
```

```
tally(~friends + sex, data = CloseFriends, margin = TRUE)
```

```
sex
friends Men Women Total
0      196  201  397
1      135  146  281
2      108  155  263
3      100  132  232
4       42   86  128
5       40   56   96
6       33   37   70
Total  654  813 1467
```

```
favstats(friends ~ sex, data = CloseFriends)
```

Exploration6.3.7

```
sex min Q1 median Q3 max      mean      sd  n missing
1  Men  0  0      1  3   6  1.860856  1.777147 654      0
2 Women 0  1      2  3   6  2.088561  1.760130 813      0
```

```
diffmean(friends ~ sex, data = CloseFriends)
```

```
diffmean
0.2277046
```

1. $H_0: \mu_{men} - \mu_{women} = 0$

$H_a: \mu_{men} - \mu_{women} \neq 0$

Test statistic: $\bar{x}_{men} - \bar{x}_{women} = -0.228$ (the difference in the sample means)

2. We simulate a world in which $\mu_{men} - \mu_{women} = 0$:

```
sim.fri <- do(1000) * diffmean(friends ~ shuffle(sex), data = CloseFriends)
head(sim.fri, 3)
```

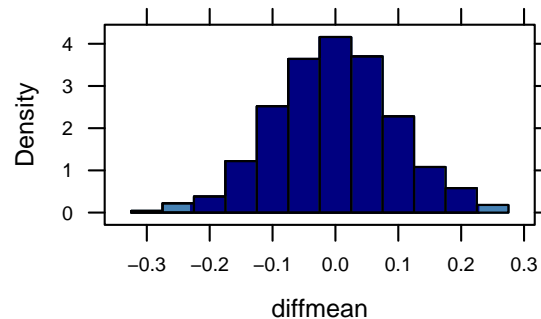
Exploration6.3.8

```
diffmean
1 -0.012333977
2 -0.153046255
3 -0.006815848
```

```

histogram(~ diffmean, data = sim.fri, width = 0.05,
          group = cut(diffmean, c(-0.4, -0.228, 0.228, 0.4)),
          fcol = c("steelblue", "navy", "steelblue"))

```



3. Strength of evidence:

```
favstats(~diffmean, data = sim.fri)
```

Exploration6.3.10

	min	Q1	median	Q3	max	mean	sd	n
	-0.2827223	-0.0626869	0.004220409	0.05940169	0.2497771	-0.0008811007	0.09430304	1000
missing	0							

```
prop(~(diffmean <= -0.228 | diffmean >= 0.228), data = sim.fri)
```

```
TRUE
0.02
```

```
t.test(friends ~ sex, data = CloseFriends)
```

Exploration6.3.13

Welch Two Sample t-test

```

data: friends by sex
t = -2.4497, df = 1392.8, p-value = 0.01442
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.41004255 -0.04536669
sample estimates:
 mean in group Men mean in group Women
      1.860856      2.088561

```

```
stat(t.test(friends ~ sex, data = CloseFriends))
```

```

t
-2.449743

```



```
pval(t.test(friends ~ sex, data = CloseFriends))
```

Exploration6.3.17

```
      p.value  
0.01441824
```

Validity Conditions

```
confint(t.test(friends ~ sex, data = CloseFriends))
```

Exploration6.3.20

mean in group Men	mean in group Women	lower	upper
1.86085627	2.08856089	-0.41004255	-0.04536669
level			
0.95000000			



Paired Data: One Quantitative Variable

7.1 Paired Designs

7.2 Simulation-Based Approach for Analyzing Paired Data

Example 7.2: Rounding First Base (continued)

Let's begin by creating a data frame that organizes this data differently. We'll call the new data frame `FirstBase2`.

```
require(tidyr)
```

Example7.2

Loading required package: `tidyr`

```
FirstBase2 <- FirstBase %>% gather(key = angle, value = time, narrow, wide)
sample(FirstBase2, 5)
```

```
      angle time orig.ids
17 narrow 5.55        17
30  wide 5.35        30
21 narrow 5.65        21
39  wide 5.35        39
3  narrow 5.60         3
```

```
head(FirstBase, 10)
```

Table7.1

```
      narrow wide
1      5.50 5.55
2      5.70 5.75
3      5.60 5.50
4      5.50 5.40
5      5.85 5.70
```

```

6    5.55 5.60
7    5.40 5.35
8    5.50 5.35
9    5.15 5.00
10   5.80 5.70

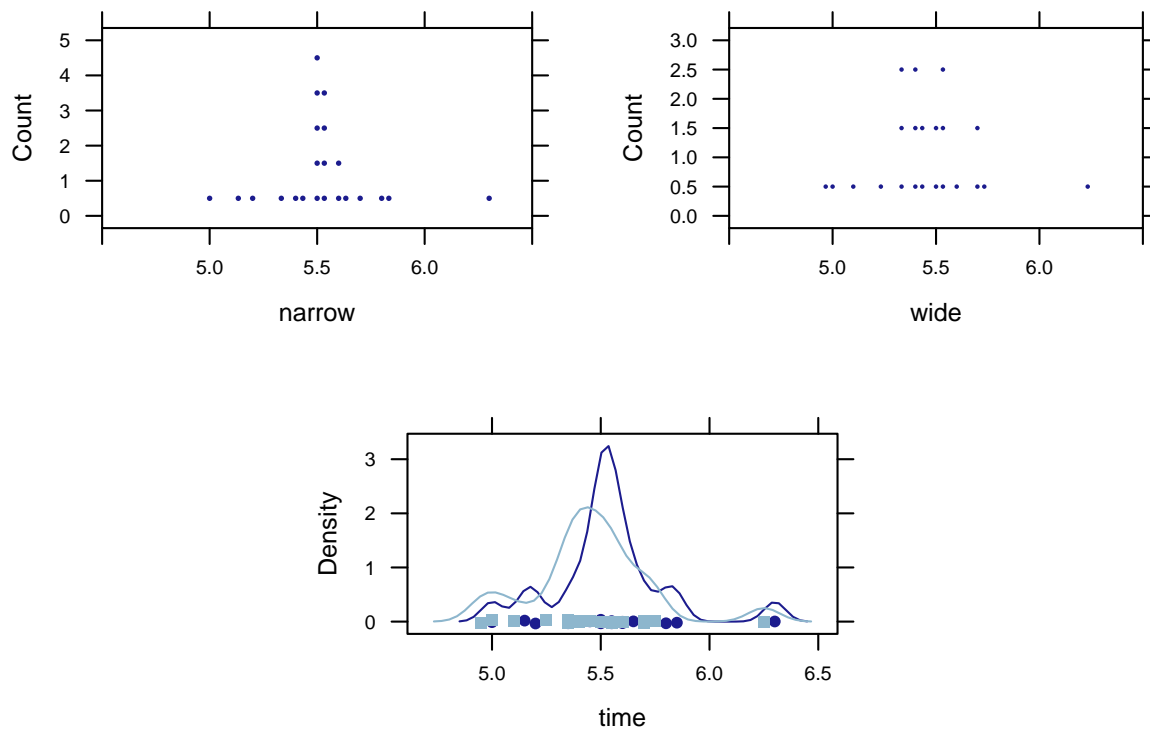
```

```

dotPlot(~narrow, data = FirstBase, nint = 40, cex = 0.2, xlim = c(4.5, 6.5))
dotPlot(~wide, data = FirstBase, nint = 40, cex = 0.1, xlim = c(4.5, 6.5))
densityplot(~time, groups = angle, data = FirstBase2)

```

Figure7.3



```
favstats(~(narrow - wide), data = FirstBase)
```

Table7.2

min	Q1	median	Q3	max	mean	sd	n	missing
-0.1	0.05	0.1	0.1375	0.2	0.075	0.08830413	22	0

```
dotPlot(~(narrow - wide), data = FirstBase)
```

Figure7.4

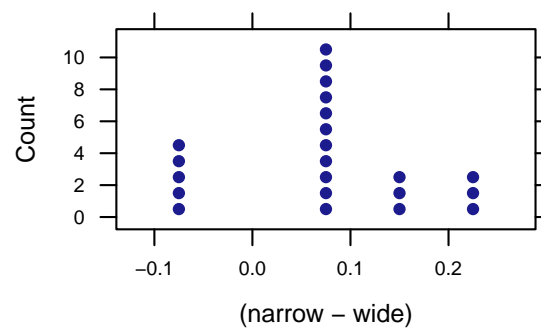


Table7.3

```
Swap.Base <- swap(FirstBase, c("narrow", "wide"))
Swap.Base
```

	narrow	wide
1	5.50	5.55
2	5.70	5.75
3	5.60	5.50
4	5.40	5.50
5	5.85	5.70
6	5.55	5.60
7	5.40	5.35
8	5.50	5.35
9	5.00	5.15
10	5.80	5.70
11	5.10	5.20
12	5.55	5.45
13	5.35	5.45
14	5.00	4.95
15	5.40	5.50
16	5.55	5.50
17	5.35	5.55
18	5.50	5.55
19	5.45	5.25
20	5.40	5.60
21	5.55	5.65
22	6.30	6.25

```
mean(~(narrow - wide), data = Swap.Base)
```

```
[1] -0.01136364
```

We simulate a world in which $\mu_d = 0$:

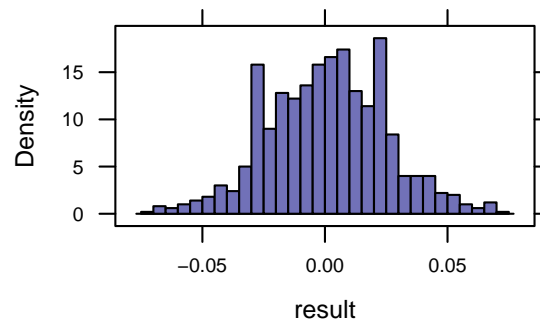
Figure7.6

```
Sim.Base <- do(1000) * mean(~ (narrow - wide), data = swap(FirstBase, c("narrow", "wide")))
head(Sim.Base, 3)
```

	result
1	0.002272727

```
2 -0.006818182
3 -0.020454545
```

```
histogram(~ result, data = Sim.Base, width = 0.005, center = 0.0025)
```



```
histogram(~result, data = Sim.Base, width = 0.005, center = 0.0025, groups = (result >= 0.075))
sd <- sd(~result, data = Sim.Base)
sd
```

Figure7.7

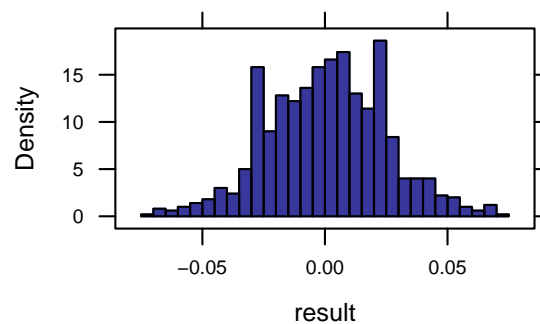
```
[1] 0.02417083
```

```
0.075 - 2 * sd
```

```
[1] 0.02665835
```

```
0.075 + 2 * sd
```

```
[1] 0.1233417
```



```
sim.firstbase <- do(1000) * diffmean(time ~ shuffle(angle), data = FirstBase2)
head(sim.firstbase, 3)
```

Figure7.8

```

      diffmean
1 -0.01590909
2  0.04318182
3  0.03409091

favstats(~diffmean, data = sim.firstbase)

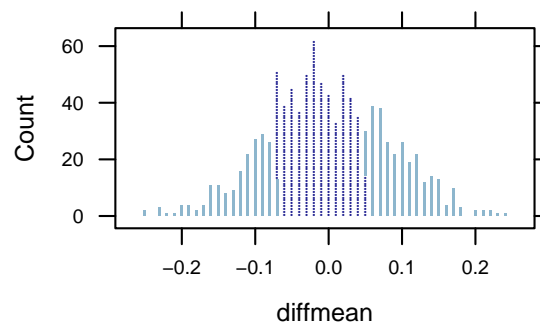
      min      Q1      median      Q3      max      mean      sd      n
-0.2522727 -0.05681818 -0.006818182  0.05681818  0.2386364 -0.002377273  0.08267672 1000
missing
      0

dotPlot(~diffmean, data = sim.firstbase, nint = 50, groups = (diffmean <= -0.075 | diffmean >=
0.05))
prop(~(diffmean <= -0.075 | diffmean >= 0.075), data = sim.firstbase)

      target level: TRUE; other levels: FALSE

TRUE
0.384

```



Exploration 7.2: Exercise and Heart Rate

```
head(JJvsBicycle)
```

Exploration7.2.5

```

JJ bicycle
1 118    118
2 146    124
3 134     92
4  94     80
5 146    111
6 114    112

```

```
favstats(~JJ, data = JJvsBicycle)
```

Exploration7.2.7

```

min      Q1 median      Q3 max      mean      sd  n missing
70 102.25    115 129.25 146 114.6364 19.5705 22      0

favstats(~bicycle, data = JJvsBicycle)

min      Q1 median      Q3 max      mean      sd  n missing
70  87.25    97.5 121.75 143 102.6818 20.65911 22      0

mean(~(JJ - bicycle), data = JJvsBicycle)

[1] 11.95455

```

```

swap.bike <- swap(JJvsBicycle, c("JJ", "bicycle"))
mean(~(JJ - bicycle), data = swap.bike)

[1] -2.954545

sd(~(JJ - bicycle), data = swap.bike)

[1] 23.33499

```

Exploration7.2.8

1. $H_0: \mu_d = 0$

$H_a: \mu_d \neq 0$

Test statistic: $\bar{x}_d = -6.773$ (the mean difference in sample)

2. We simulate a world in which $\mu_d = 0$:

```

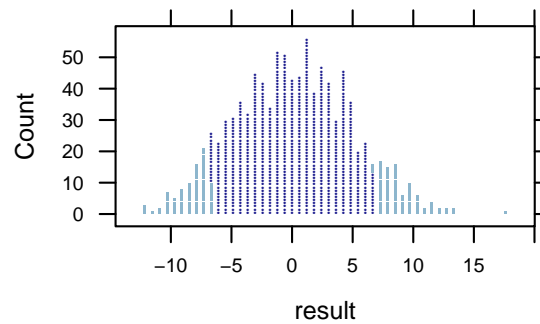
sim.bike <- do(1000) * mean(~ (JJ - bicycle),
                             data = swap(JJvsBicycle, c("JJ", "bicycle")))
head(sim.bike, 3)

      result
1 -4.045455
2 -3.136364
3 -3.136364

dotPlot(~ result, data = sim.bike, nint = 50, groups = (result <=-6.773 | result >=6.773))

```

Exploration7.2.10



3. Strength of evidence:

```
favstats(~result, data = sim.bike)
```

Exploration7.2.12

	min	Q1	median	Q3	max	mean	sd	n	missing
	-12.22727	-3.409091	0.04545455	3.431818	17.5	0.07590909	4.826593	1000	0

```
prop(~(result <= -6.773 | result >= 6.773), data = sim.bike)
```

```
TRUE
0.169
```

Standardized statistic:

```
sd <- sd(~result, data = sim.bike)
xpnorm(-6.773, 0, sd, plot = FALSE)
```

Exploration7.2.14

If $X \sim N(0, 4.82659265076836)$, then

```
P(X <= -6.773) = P(Z <= -1.403) = 0.0803
P(X > -6.773) = P(Z > -1.403) = 0.9197
[1] 0.08026856
```

95% confidence interval using 2SD Method:

```
sd <- sd(~result, data = sim.bike)
-6.773 - 2 * sd
```

Exploration7.2.15

```
[1] -16.42619
```

```
-6.773 + 2 * sd
```

```
[1] 2.880185
```

Let's again create the stacked data.

```
require(tidyr)
JJvsBicycle2 <- JJvsBicycle %>% gather(key = exercise, value = heartrate, JJ:bicycle)
sample(JJvsBicycle2, 5)
```

Exploration7.2.17

	exercise	heartrate	orig.ids
32	bicycle	82	32
27	bicycle	111	27
1	JJ	118	1
12	JJ	99	12
25	bicycle	92	25

```
sim.bike2 <- do(1000) * diffmean(heartrate ~ shuffle(exercise), data = JJvsBicycle2)
head(sim.bike2, 3)
```

Exploration7.2.17b

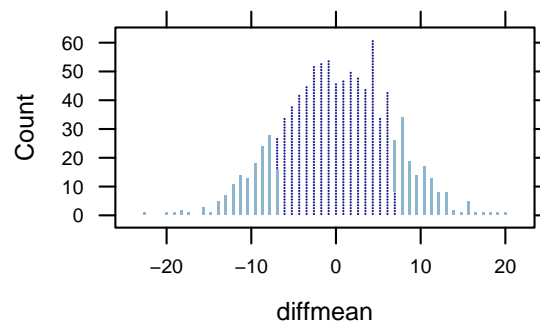
	diffmean
1	-8.95454545
2	-12.50000000
3	-0.04545455

```
favstats(~diffmean, data = sim.bike2)
```

min	Q1	median	Q3	max	mean	sd	n	missing
-22.86364	-4.25	0.04545455	4.409091	19.68182	0.02490909	6.401086	1000	0

```
dotPlot(~diffmean, data = sim.bike2, nint = 50, groups = (diffmean <= -6.773 | diffmean >=
6.773))
prop(~(diffmean <= -6.773 | diffmean >= 6.773), data = sim.bike2)
```

```
TRUE
0.29
```



7.3 Theory-Based Approach to Analyzing Data from Paired Samples

Example 7.3: How Many M&Ms Would You Like?

```
head(BowlsMMs)
```

Table7.4

	small	large
1	33	41
2	24	92
3	35	61
4	24	19
5	40	21
6	33	35

```
favstats(~small, data = BowlsMMs)
```

Table7.5

min	Q1	median	Q3	max	mean	sd	n	missing
24	26	34	40	88	38.58824	16.89696	17	0

```
favstats(~large, data = BowlsMMs)
```

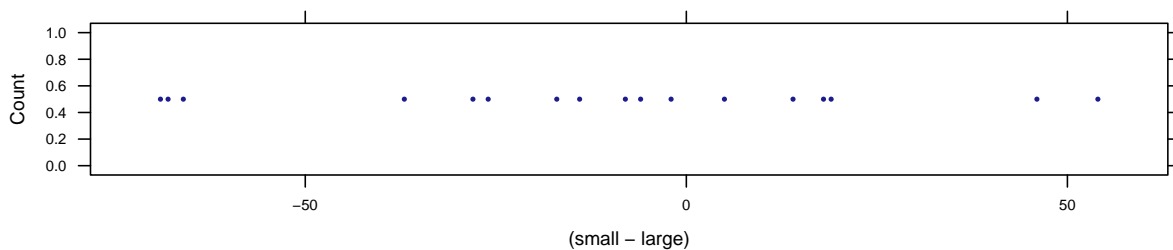
min	Q1	median	Q3	max	mean	sd	n	missing
11	33	42	62	104	49.47059	27.20781	17	0

```
favstats(~(small - large), data = BowlsMMs)
```

min	Q1	median	Q3	max	mean	sd	n	missing
-69	-28	-8	14	54	-10.88235	36.30062	17	0

```
dotPlot(~(small - large), data = BowlsMMs, width = 1, cex = 0.05)
```

Figure7.9



1. $H_0: \mu_d = 0$

$H_a: \mu_d < 0$

Test statistic: $\bar{x}_d = -10.88$ (the mean difference in paired samples)

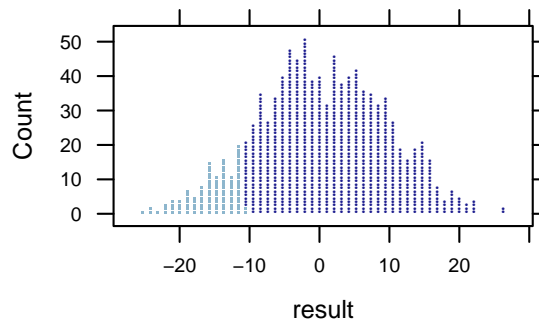
2. We simulate a world in which $\mu_d = 0$:

```
sim.mm <- do(1000) * mean(~ (small - large), data = swap(BowlsMMs, c("small", "large")))
head(sim.mm, 3)
```

Figure7.10

```
      result
1 -2.294118
2 -1.352941
3 -9.941176
```

```
dotPlot(~ result, data = sim.mm, nint = 50, groups = (result <= -10.88))
```



3. Strength of evidence:

```
favstats(~result, data = sim.mm)
```

Figure7.11

```
      min      Q1    median      Q3      max      mean      sd  n missing
-24.76471 -5.823529 0.1176471 6.882353 26.76471 0.3254118 9.132693 1000      0
```

```
prop(~(result <= -10.88), data = sim.mm)
```

```
TRUE
0.11
```

Theory-based approach

```
t.test(small, large, data = BowlsMMs, paired = TRUE, alt = "less")
```

Figure7.12

Paired t-test

data: small and large

t = -1.236, df = 16, p-value = 0.1171

alternative hypothesis: true difference in means is less than 0

95 percent confidence interval:

-Inf 4.488747

sample estimates:

mean of the differences

-10.88235

Exploration 7.3: comparing Auction Formats

```
head(Auction)
```

Exploration7.3.1

	dutch	FP
1	25	26.25
2	24	25.25
3	26	27.00
4	20	20.75
5	20	20.75
6	15	15.25

```
summary(Auction)
```

Exploration7.3.5

dutch		FP	
Min.	: 0.150	Min.	: 0.100
1st Qu.:	2.000	1st Qu.:	1.188
Median :	3.000	Median :	2.275
Mean :	5.162	Mean :	4.779
3rd Qu.:	7.000	3rd Qu.:	6.050
Max.	:26.000	Max.	:27.000

```
favstats(~(dutch - FP), data = Auction)
```

min	Q1	median	Q3	max	mean	sd	n	missing
-1.25	0	0.25	0.5	2.4	0.3835227	0.6752063	88	0

1. $H_0: \mu_d = 0$

$H_a: \mu_d \neq 0$

Test statistic: $\bar{x}_d = 0.384$ (the mean difference in paired samples)

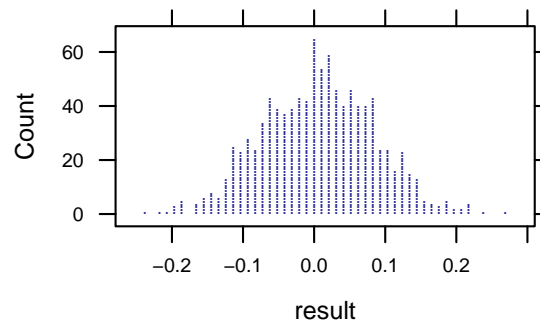
2. We simulate a world in which $\mu_d = 0$:

```
sim.auction <- do(1000) * mean(~ (dutch - FP), data = swap(Auction, c("dutch", "FP")))
head(sim.auction, 3)
```

Exploration7.3.5b

	result
1	0.02897727
2	0.02556818
3	-0.06988636

```
dotPlot(~ result, data = sim.auction, groups = (result <= -0.384 | result >= 0.384), nint = 50)
```



3. Strength of evidence:

```
favstats(~result, data = sim.auction)
```

Exploration7.3.5c

	min	Q1	median	Q3	max	mean	sd	n
	-0.2414773	-0.05170455	0.007386364	0.05965909	0.2653409	0.005969318	0.07851641	1000
missing	0							

```
prop(~(result <= -0.384 | result >= 0.384), data = sim.auction)
```

```
TRUE
0
```

4. t-test for paired samples (theory-based approach):

```
t.test(Auction$dutch, Auction$FP, paired = TRUE)
```

Exploration7.3.7

Paired t-test

```
data: Auction$dutch and Auction$FP
t = 5.3284, df = 87, p-value = 7.692e-07
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.2404602 0.5265853
sample estimates:
mean of the differences
      0.3835227
```

```
t.test(~(dutch - FP), data = Auction)
```

One Sample t-test

```
data: data$(dutch - FP)
t = 5.3284, df = 87, p-value = 7.692e-07
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 0.2404602 0.5265853
sample estimates:
mean of x
      0.3835227
```

95% confidence interval using the t-test:

```
confint(t.test(Auction$dutch, Auction$FP, paired = TRUE))
```

Exploration7.3.8

mean of the differences	lower	upper
0.3835227	0.2404602	0.5265853
level		
0.9500000		

8

Comparing More Than Two Proportions

8.1 Simulation-Based Approach to Compare Multiple Proportions

Example 8.1: Coming to a Stop

```
require(vcd)
sample(Stop, 5)
```

Table 8.1

```
      position stop orig.ids
215    lead    no      215
166   single    no      166
238   follow   yes      238
300   follow    no      300
147   single   yes      147
```

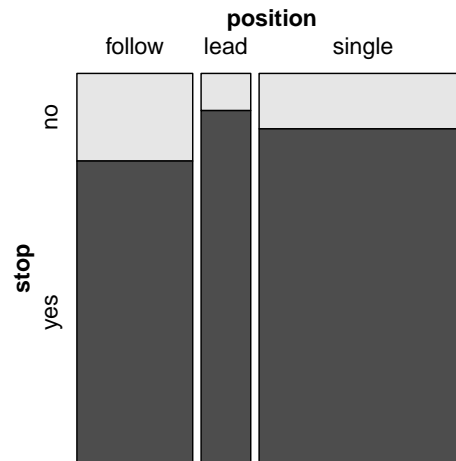
```
tally(~stop + position, data = Stop, margins = TRUE)
```

```
      position
stop   follow lead single Total
no      22    4    25    51
yes     76   38   151   265
Total   98   42   176   316
```

```
tally(stop ~ position, data = Stop)
```

```
      position
stop   follow lead single
no      22    4    25
yes     76   38   151
```

```
mosaic(stop ~ position, data = Stop, direction = "v")
```



Mean Absolute Difference (MAD)

We can input the proportions to compute MAD:

```
MAD(prop(stop ~ position, data = Stop))
```

Figure8.1

```
[1] 0.0861678
```

Then we can shuffle the response variable:

```
MAD(prop(shuffle(stop) ~ position, data = Stop))
```

Figure8.2

```
[1] 0.03834261
```

1. $H_0: \pi_{Single} = \pi_{Lead} = \pi_{Follow}$

H_a : At least one of the three long-run probabilities is different from the others

Test statistic: $MAD = 0.086$ (the absolute mean difference)

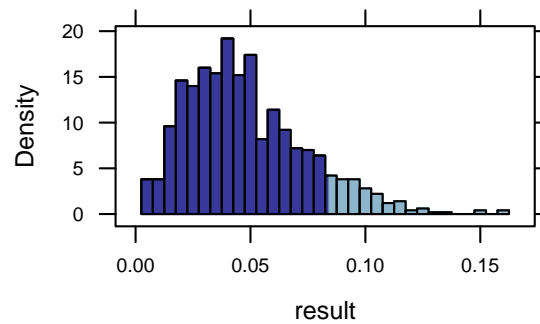
2. We simulate a world in which $MAD = 0$:

```
sim.stop <- do(1000) * MAD(prop(shuffle(stop) ~ position, data = Stop))
head(sim.stop, 3)
```

Figure8.3

```
      result
1 0.01839827
2 0.04336735
3 0.08390023
```

```
histogram(~result, data = sim.stop, width = 0.005, groups = (result >= 0.086))
```



3. Strength of evidence:

```
favstats(~result, data = sim.stop)
```

Figure8.3b

	min	Q1	median	Q3	max	mean	sd	n
0.005050505	0.02898887	0.04336735	0.06349206	0.1609977	0.04788469	0.02602633	1000	
missing	0							

```
prop(~(result >= 0.086), data = sim.stop)
```

```
TRUE
0.1
```

Exploration 8.1: Recruiting Organ Donors

```
head(OrganDonor)
```

Exploration8.1.1

```
default choice
1 opt-in donor
2 opt-in donor
3 opt-in donor
4 opt-in donor
5 opt-in donor
6 opt-in donor
```

```
tally(~choice + default, data = OrganDonor)
```

Exploration8.1.5

	default		
choice	neutral	opt-in	opt-out
donor	44	23	41
not	12	32	9

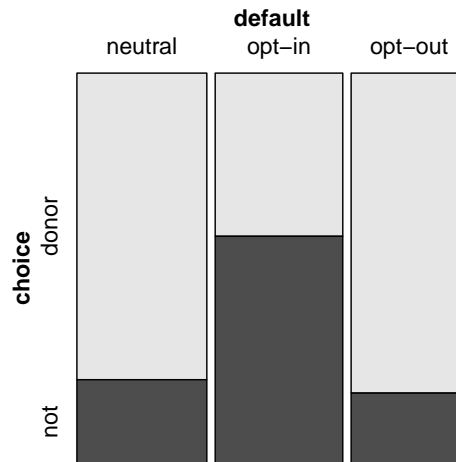
```
tally(choice ~ default, data = OrganDonor)
```

```

      default
choice neutral opt-in opt-out
donor    44    23    41
not      12    32    9

```

```
mosaic(choice ~ default, data = OrganDonor, direction = "v")
```



```
MAD(prop(choice ~ default, data = OrganDonor))
```

Exploration8.1.9

```
[1] 0.2678788
```

1. $H_0: \pi_{opt-in} = \pi_{opt-out} = \pi_{neutral}$

H_a : At least one of the three long-run probabilities is different from the others

Test statistic: $MAD = 0.268$ (the absolute mean difference)

2. We simulate a world in which $MAD = 0$:

```
sim.donor <- do(1000) * MAD(prop(shuffle(choice) ~ default, data = OrganDonor))
head(sim.donor, 3)
```

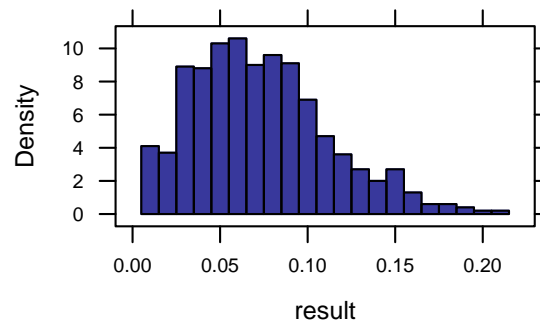
Exploration8.1.11

```

      result
1 0.13212121
2 0.11190476
3 0.09212121

```

```
histogram(~result, data = sim.donor, width = 0.01, groups = (result >= 0.268))
```



3. Strength of evidence:

```
favstats(~result, data = sim.donor)
```

Exploration8.1.12

	min	Q1	median	Q3	max	mean	sd	n	missing
	0.01238095	0.04415584	0.06818182	0.0969697	0.2121212	0.07368472	0.03865989	1000	0

```
prop(~(result >= 0.086), data = sim.stop)
```

```
TRUE
0.1
```

8.2 Theory-Based Approach to Compare Multiple Proportions

Example 8.2: Sham Acupuncture

```
sample(Acupuncture, 5)
```

Table8.2

	acupuncture	improvement	orig.ids
335	Sham	Better	335
444	None	Better	444
352	Sham	Better	352
1005	None	Not	1005
433	None	Better	433

```
tally(~improvement + acupuncture, data = Acupuncture, margins = TRUE)
```

	acupuncture			
improvement	None	Real	Sham	Total
Better	106	184	171	461
Not	282	203	216	701
Total	388	387	387	1162

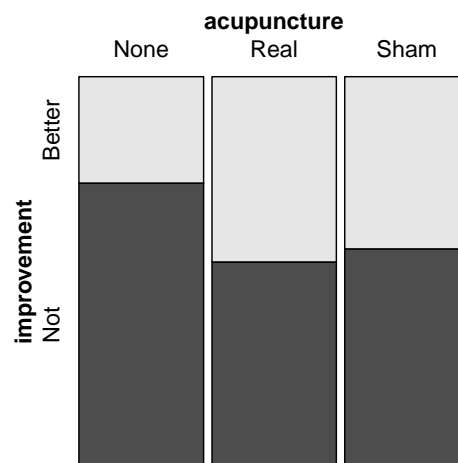
```
tally(improvement ~ acupuncture, data = Acupuncture)
```

	acupuncture		
improvement	None	Real	Sham
Better	106	184	171
Not	282	203	216

```
mosaic(improvement ~ acupuncture, data = Acupuncture, direction = "v")
MAD(prop(improvement ~ acupuncture, data = Acupuncture))
```

Figure8.4

```
[1] 0.1348375
```



1. $H_0: \pi_{real} = \pi_{sham} = \pi_{none}$

H_a : At least one of the three long-run probabilities is different from the others

Test statistic: $MAD = 0.135$ (the absolute mean difference)

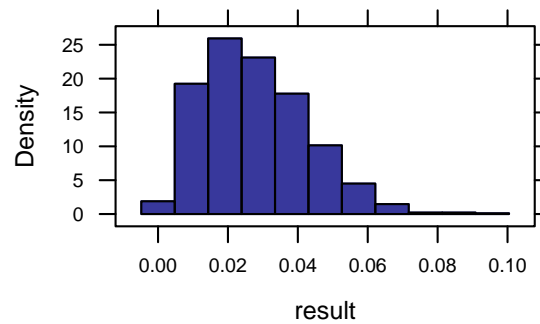
2. We simulate a world in which $MAD = 0$:

```
sim.acu <- do(1000) * MAD(prop(shuffle(improvement) ~ acupuncture, data = Acupuncture))
head(sim.acu, 3)
```

Figure8.5

```
      result
1 0.02583979
2 0.01961516
3 0.01307973
```

```
histogram(~result, data = sim.acu, groups = (result >= 0.135))
```



3. Strength of evidence:

```
favstats(~result, data = sim.acu)
```

Figure8.5b

```
      min      Q1    median      Q3      max      mean      sd      n
0.001722653 0.01616985 0.02583979 0.03715247 0.09736541 0.02773222 0.01474666 1000
missing
0
```

```
prop(~(result >= 0.135), data = sim.acu)
```

```
TRUE
0
```

Theory-based approach: The chi-square test

For the chi-square test, data must be tabulated.

```
acu.table <- tally(~improvement + acupuncture, data = Acupuncture)
acu.table
```

Figure8.7

```
      acupuncture
improvement None Real Sham
Better    106   184   171
Not       282   203   216
```

```
chisq.test(acu.table)
```

Pearson's Chi-squared test

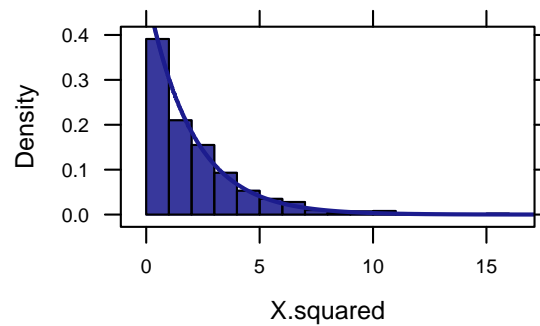
```
data: acu.table
X-squared = 38.054, df = 2, p-value = 5.453e-09
```

```
sim.acuX2 <- do(1000) * chisq.test(tally(~shuffle(improvement) + acupuncture, data = Acupuncture))$statistic
head(sim.acuX2, 3)
```

Figure8.7b

```
X.squared
1  1.347661
2  3.420586
3  2.489779

histogram(~X.squared, data = sim.acuX2, width = 1, center = 0.5, groups = X.squared >= 38.05)
plotDist("chisq", df = 2, add = TRUE)
```



```
xchisq.test(acu.table) # with cell contributions and expected counts
```

Figure8.8

Pearson's Chi-squared test

```
data: x
X-squared = 38.054, df = 2, p-value = 5.453e-09
```

```
106      184      171
(153.93) (153.53) (153.53)
[14.92] [ 6.05] [ 1.99]
<-3.86> < 2.46> < 1.41>
```

```
282      203      216
(234.07) (233.47) (233.47)
[ 9.82] [ 3.98] [ 1.31]
< 3.13> <-1.99> <-1.14>
```

```
key:
observed
(expected)
[contribution to X-squared]
<residual>
```

Exploration 8.2: Conserving Hotel Towels

```
head(Towels)
```

Exploration8.2.2

```
treatment
```


towel	none	samerm	citizen	gender	guest
reuse	113	151	145	127	150
not reuse	192	155	189	183	190

Here, we can see that the data set is already in table format. But let's also store it as a data frame for future use.

Towels

	treatment				
towel	none	samerm	citizen	gender	guest
reuse	113	151	145	127	150
not reuse	192	155	189	183	190

```
Towels1 <- rbind(
  do(113) * data.frame(treatment = "none", towel = "reuse"),
  do(192) * data.frame(treatment = "none", towel = "not"),
  do(151) * data.frame(treatment = "samerm", towel = "reuse"),
  do(155) * data.frame(treatment = "samerm", towel = "not"),
  do(145) * data.frame(treatment = "citizen", towel = "reuse"),
  do(189) * data.frame(treatment = "citizen", towel = "not"),
  do(127) * data.frame(treatment = "gender", towel = "reuse"),
  do(183) * data.frame(treatment = "gender", towel = "not"),
  do(150) * data.frame(treatment = "guest", towel = "reuse"),
  do(190) * data.frame(treatment = "guest", towel = "not")
)
```

```
prop.table(Towels, margin = 2)
```

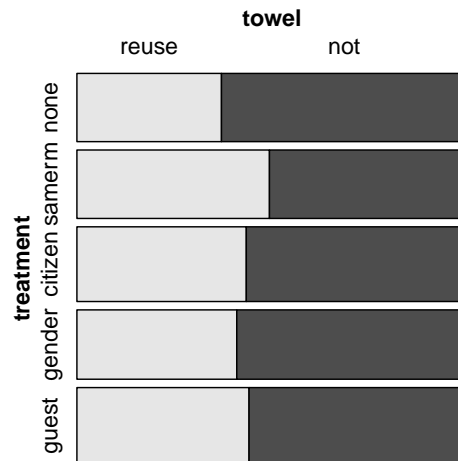
	treatment				
towel	none	samerm	citizen	gender	guest
reuse	0.3704918	0.4934641	0.4341317	0.4096774	0.4411765
not reuse	0.6295082	0.5065359	0.5658683	0.5903226	0.5588235

```
tally(towel ~ treatment, data = Towels1)
```

	treatment				
towel	none	samerm	citizen	gender	guest
reuse	113	151	145	127	150
not	192	155	189	183	190

```
mosaic(towel ~ treatment, data = Towels1)
```

Exploration8.2.5



Exploration8.2.6

```
MAD(prop(towel ~ treatment, data = Towels1))
```

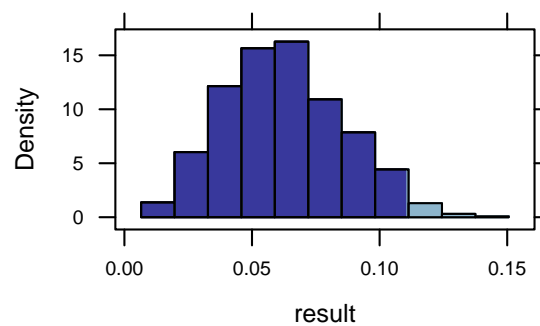
```
[1] 0.1109774
```

```
sim.towel <- do(1000) * MAD(prop(shuffle(towel) ~ treatment, data = Towels1))
head(sim.towel, 3)
```

```
      result
1 0.00687602
2 0.05258570
3 0.04466305
```

```
histogram(~result, data = sim.towel, groups = (result >= 0.111))
prop(~(result >= 0.111), data = sim.towel)
```

```
TRUE
0.024
```



```
prop(~towel, data = Towels1)
```

Exploration8.2.7

```
reuse
0.430094
```

```
chisq.test(Towels)
```

Exploration8.2.13

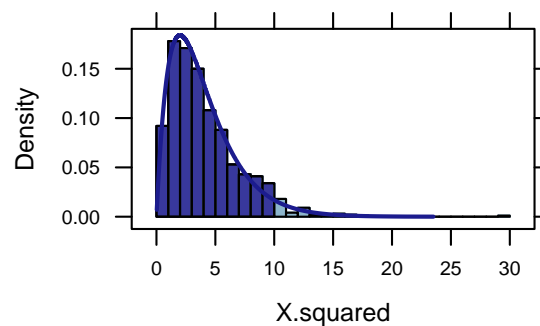
Pearson's Chi-squared test

```
data: Towels
X-squared = 10.153, df = 4, p-value = 0.03792
```

```
sim.towelX2 <- do(1000) * chisq.test(tally(~shuffle(towel) + treatment, data = Towels1))$statistic
head(sim.towelX2, 3)
```

```
X.squared
1  7.782173
2  2.882825
3  2.127759
```

```
histogram(~X.squared, data = sim.towelX2, width = 1, center = 0.5, groups = X.squared >= 10.153)
plotDist("chisq", df = 4, add = TRUE)
```



```
xchisq.test(Towels)
```

Exploration8.2.15

Pearson's Chi-squared test

```
data: x
X-squared = 10.153, df = 4, p-value = 0.03792
```

```
113      151      145      127      150
(131.18) (131.61) (143.65) (133.33) (146.23)
```

```
[2.5192] [2.8571] [0.0127] [0.3004] [0.0971]
<-1.587> < 1.690> < 0.113> <-0.548> < 0.312>

      192      155      189      183      190
(173.82) (174.39) (190.35) (176.67) (193.77)
[1.9012] [2.1562] [0.0096] [0.2267] [0.0733]
< 1.379> <-1.468> <-0.098> < 0.476> <-0.271>

key:
observed
(expected)
[contribution to X-squared]
<residual>
```

Follow-up Analysis

Exploration 8.2b: Near-sightedness and Nightlights revisited

NightLight1

Exploration8.2b

	Darkness	NightLight	RoomLight
Near	18	78	41
Not	154	154	34

Alternative formula for chi-square statistic

```
xchisq.test(NightLight1)
```

Exploration8.2b.4

Pearson's Chi-squared test

```
data: x
X-squared = 55.519, df = 2, p-value = 8.795e-13
```

```
      18      78      41
( 49.19) ( 66.35) ( 21.45)
[19.78] [ 2.04] [17.82]
<-4.45> < 1.43> < 4.22>
```

```
     154     154      34
(122.81) (165.65) ( 53.55)
[ 7.92] [ 0.82] [ 7.14]
< 2.81> <-0.90> <-2.67>
```

```
key:
observed
(expected)
[contribution to X-squared]
<residual>
```

We can see that `NightLight1` is in table format. Let's create new data frame it for some easier analysis.

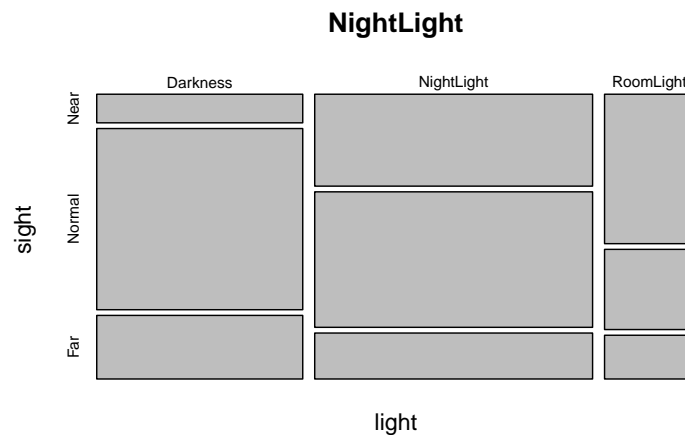
NightLight2

	light		
sight	Darkness	NightLight	RoomLight
Nearsighted	18	78	41
Normal	114	115	22
Farsighted	40	39	12

```
NightLight <- rbind(
  do(18) * data.frame(light = "Darkness", sight = "Near"),
  do(114) * data.frame(light = "Darkness", sight = "Normal"),
  do(40) * data.frame(light = "Darkness", sight = "Far"),
  do(78) * data.frame(light = "NightLight", sight = "Near"),
  do(115) * data.frame(light = "NightLight", sight = "Normal"),
  do(39) * data.frame(light = "NightLight", sight = "Far"),
  do(41) * data.frame(light = "RoomLight", sight = "Near"),
  do(22) * data.frame(light = "RoomLight", sight = "Normal"),
  do(12) * data.frame(light = "RoomLight", sight = "Far")
)
```

```
mosaicplot(light ~ sight, data = NightLight)
```

Exploration8.2b.7



```
chisq.test(tally(~sight + light, data = NightLight))
```

Exploration8.2b.10

Pearson's Chi-squared test

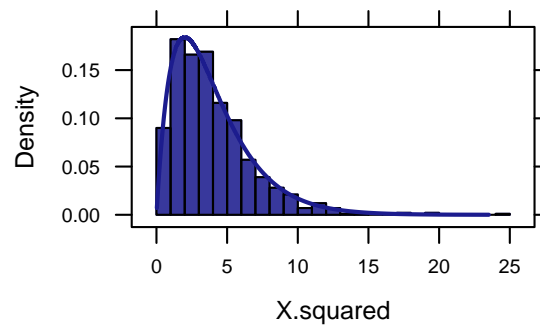
```
data:  tally(~sight + light, data = NightLight)
X-squared = 56.513, df = 4, p-value = 1.565e-11
```

Exploration8.2b.11

```
sim.nightX2 <- do(1000) * chisq.test(tally(~shuffle(light) + sight, data = NightLight))$statistic
head(sim.nightX2, 3)
```

```
  X.squared
1  2.383670
2  1.775620
3  1.571492
```

```
histogram(~X.squared, data = sim.nightX2, width = 1, center = 0.5, groups = X.squared >= 56.514)
plotDist("chisq", df = 4, add = TRUE)
```



Exploration8.2b.12

```
xchisq.test(NightLight2)
```

Pearson's Chi-squared test

```
data: x
X-squared = 56.513, df = 4, p-value = 1.565e-11
```

```
  18      78      41
( 49.19) ( 66.35) ( 21.45)
[ 19.78] [  2.04] [ 17.82]
<-4.45> <  1.43> <  4.22>
```

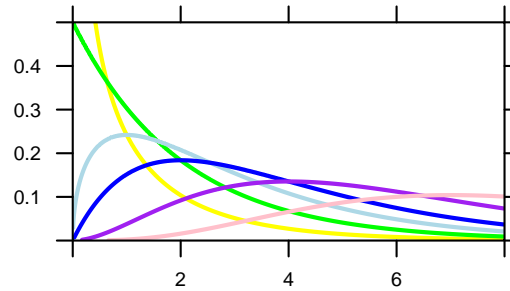
```
  114      115      22
( 90.13) (121.57) ( 39.30)
[  6.32] [  0.36] [  7.62]
<  2.51> <-0.60> <-2.76>
```

```
  40      39      12
( 32.68) ( 44.08) ( 14.25)
[  1.64] [  0.58] [  0.35]
<  1.28> <-0.76> <-0.60>
```

```
key:
observed
(expected)
[contribution to X-squared]
<residual>
```

```
plotDist("chisq", params = list(df = 1), col = "yellow", ylim = c(0, 0.5), xlim = c(0, 8))
plotDist("chisq", params = list(df = 2), col = "green", add = TRUE)
plotDist("chisq", params = list(df = 3), col = "lightblue", add = TRUE)
plotDist("chisq", params = list(df = 4), col = "blue", add = TRUE)
plotDist("chisq", params = list(df = 6), col = "purple", add = TRUE)
plotDist("chisq", params = list(df = 9), col = "pink", add = TRUE)
```

Figure8.9



9

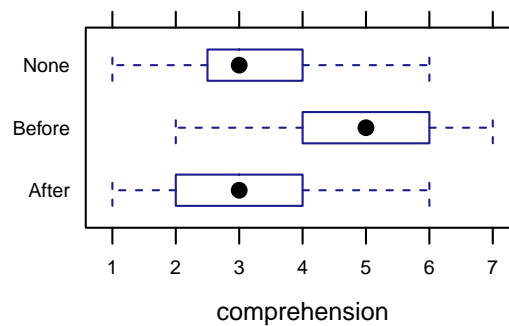
Comparing More than Two Means

9.1 Simulation-Based Approach for Comparing More than Two Groups with a Quantitative Response

Example 9.1: Comprehending Ambiguous Prose

```
bwplot(condition ~ comprehension, data = Comprehension, horizontal = TRUE)
```

Figure9.2



```
favstats(comprehension ~ condition, data = Comprehension)
```

Table9.1

	condition	min	Q1	median	Q3	max	mean	sd	n	missing
1	After	1	2.0	3	4	6	3.210526	1.397575	19	0
2	Before	2	4.0	5	6	7	4.947368	1.311220	19	0
3	None	1	2.5	3	4	6	3.368421	1.256562	19	0

```
MAD(mean(comprehension ~ condition, data = Comprehension))
```

Figure9.3

```
[1] 1.157895
```

1. $H_0: \pi_{After} = \pi_{Before} = \pi_{None}$

H_a : At least one of the three long-run probabilities is different from the others

Test statistic: $MAD = 1.16$ (the mean absolute difference)

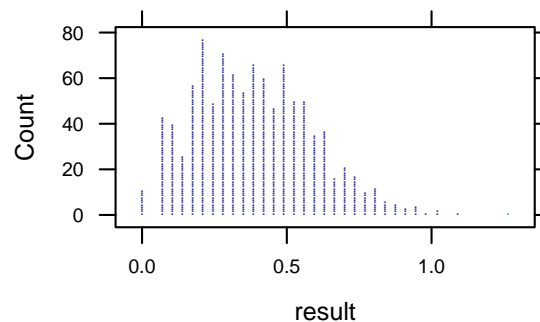
2. We simulate a world in which $MAD = 0$:

```
sim.comp <- do(1000) * MAD(mean(shuffle(comprehension) ~ condition, data = Comprehension))
head(sim.comp, 3)
```

Figure9.3b

```
      result
1 0.3859649
2 0.5263158
3 0.4912281
```

```
dotPlot(~result, data = sim.comp, width = 0.005, groups = (result >= 1.16))
```



3. Strength of evidence:

```
favstats(~result, data = sim.comp)
```

Figure9.3c

```
min      Q1    median      Q3      max      mean      sd    n missing
0 0.2105263 0.3859649 0.5263158 1.263158 0.3874737 0.2026738 1000      0
```

```
prop(~(result >= 1.16), data = sim.comp)
```

```
TRUE
0.001
```

Exploration 9.1: Exercise and Brain Volume

```
head(Brain)
```

Exploration9.1.3

```
  treatment brain_change
1   TaiChi      0.987
2   TaiChi      1.960
3   TaiChi      0.304
4   TaiChi      0.005
5   TaiChi     -1.829
6   TaiChi      1.227
```

```
favstats(brain_change ~ treatment, data = Brain)
```

Exploration9.1.6

	treatment	min	Q1	median	Q3	max	mean	sd	n	missing
1	None	-2.034	-1.16875	-0.585	0.9725	2.011	-0.2401250	1.2584309	24	0
2	Social	-1.359	0.00750	0.596	0.8060	1.796	0.4056296	0.6968969	27	0
3	TaiChi	-1.829	0.00500	0.449	0.9870	2.201	0.4710690	0.8557466	29	0
4	Walking	-3.470	-1.05850	-0.026	0.9710	1.833	-0.1503333	1.3868388	27	0

```
MAD(mean(brain_change ~ treatment, data = Brain))
```

Exploration9.1.13

```
[1] 0.6723862
```

```
MAD(mean(shuffle(brain_change) ~ treatment, data = Brain))
```

Exploration9.1.16

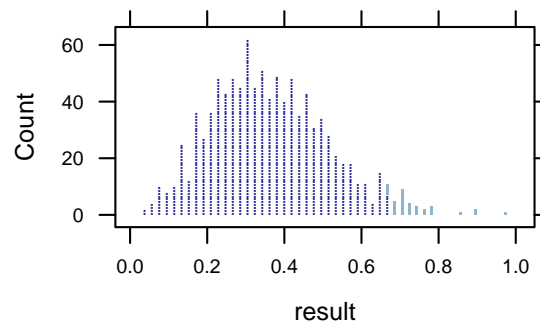
```
[1] 0.4693284
```

```
sim.brain <- do(1000) * MAD(mean(shuffle(brain_change) ~ treatment, data = Brain))
head(sim.brain, 3)
```

Exploration9.1.19

```
      result
1 0.3482152
2 0.2412166
3 0.4617909
```

```
dotPlot(~result, data = sim.brain, n = 50, groups = (result >= 0.672))
```



```
prop(~(result >= 0.672), data = sim.brain)
```

Exploration9.1.20

```
TRUE
0.034
```

```
sim.10000 <- do(10000) * MAD(mean(shuffle(brain_change) ~ treatment, data = Brain))
head(sim.10000, 3)
```

Exploration9.1.20b

```
      result
1 0.4116506
2 0.2878076
3 0.3137294
```

```
prop(~(result >= 0.672), data = sim.10000)
```

```
target level: TRUE; other levels: FALSE
```

```
TRUE
0.0325
```

9.2 Theory-based Approach to Comparing More than Two Groups with a Quantitative Response

Example 9.2: Recalling Ambiguous Prose

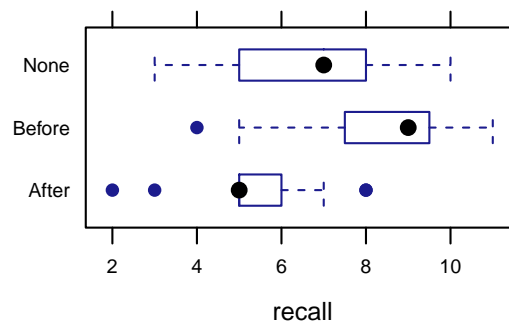
```
bwplot(condition ~ recall, data = Recall, horizontal = TRUE)
favstats(recall ~ condition, data = Recall)
```

Figure9.4

	condition	min	Q1	median	Q3	max	mean	sd	n	missing
1	After	2	5.0	5	6.0	8	5.368421	1.460994	19	0
2	Before	4	7.5	9	9.5	11	8.263158	1.820931	19	0
3	None	3	5.0	7	8.0	10	6.631579	2.005839	19	0

```
MAD(mean(recall ~ condition, data = Recall))
```

```
[1] 1.929825
```



1. $H_0: \pi_{After} = \pi_{Before} = \pi_{None}$

H_a : At least one of the three long-run probabilities is different from the others

Test statistic: $MAD = 1.93$ (the mean absolute difference)

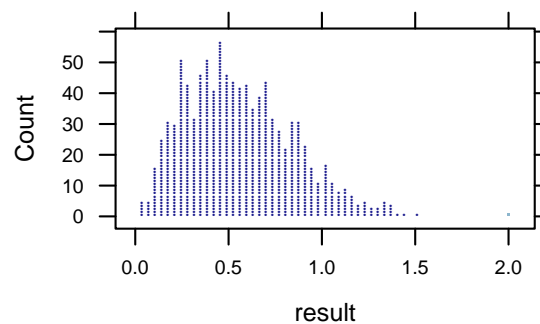
2. We simulate a world in which $MAD = 0$:

```
sim.recall <- do(1000) * MAD(mean(shuffle(recall) ~ condition, data = Recall))
head(sim.recall, 3)
```

Figure9.5

```
      result
1 0.8421053
2 0.5964912
3 0.1403509
```

```
dotPlot(~result, data = sim.recall, width = 0.005, groups = (result >= 1.93))
```



3. Strength of evidence:

```
favstats(~result, data = sim.recall)
```

Figure9.5b

```
      min      Q1    median      Q3 max      mean      sd  n missing
0.03508772 0.3508772 0.5263158 0.7368421 2 0.5612982 0.2880023 1000      0
```

```
prop(~(result >= 1.93), data = sim.recall)
```

```
TRUE
0.001
```

```
sim.recallF <- do(1000) * anova(lm(shuffle(recall) ~ condition, data = Recall))
head(sim.recallF, 3)
```

Figure9.8

```
      source df      SS      MS      F      pval .row .index
condition condition 2  10.77193  5.385965  1.212906  0.3053025   1   1
Residuals Residuals 54 239.78947  4.440546      NA      NA   2   1
condition1 condition 2  10.77193  5.385965  1.212906  0.3053025   1   2
```

```
histogram(~F, data = sim.recallF, n = 25)
prop(~(F >= 12.67), data = sim.recallF)
```

target level: TRUE; other levels: FALSE

TRUE
0

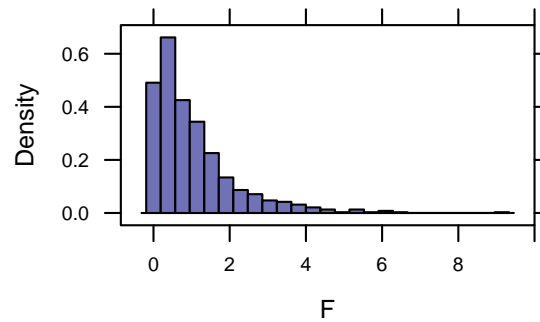


Figure9.9

```
histogram(~F, data = sim.recallF, n = 25)
plotDist("f", df1 = 2, df2 = 52, add = TRUE)
```

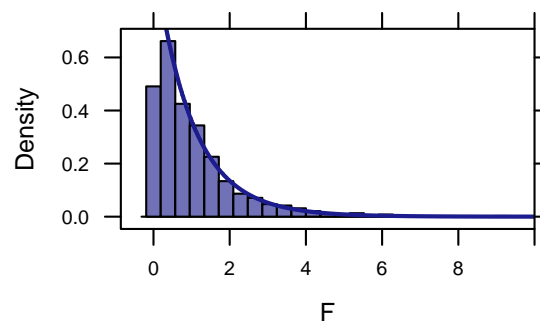


Figure9.10

```
anova(lm(recall ~ condition, data = Recall))
```

Analysis of Variance Table

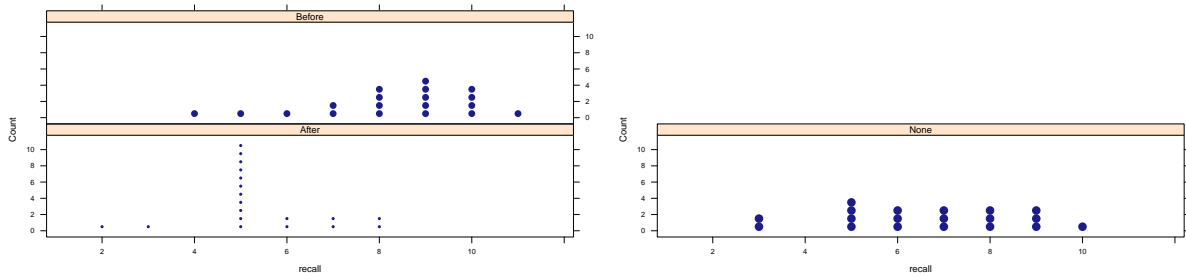
Response: recall

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
condition	2	80.035	40.018	12.672	3.074e-05 ***
Residuals	54	170.526	3.158		

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Figure9.11

```
dotPlot(~recall | condition, data = Recall, cex = 0.5, width = 1, layout = c(1, 2))
```



```
confint(lm(recall ~ condition, data = Recall))
```

Example9.2

```

              2.5 %   97.5 %
(Intercept)  4.5510669 6.185775
conditionBefore 1.7388236 4.050650
conditionNone  0.1072446 2.419071

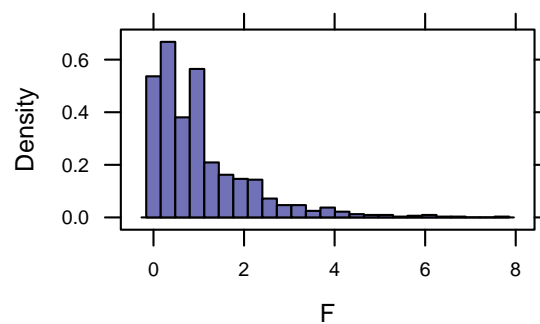
```

```
sim.compF <- do(1000) * anova(lm(shuffle(comprehension) ~ condition, data = Comprehension))
head(sim.compF, 3)
```

Figure9.12

	source	df	SS	MS	F	pval	.row	.index
condition	condition	2	0.1052632	0.05263158	0.02195122	0.9782967	1	1
Residuals	Residuals	54	129.4736842	2.39766082	NA	NA	2	1
condition1	condition	2	8.8421053	4.42105263	1.97733217	0.1483354	1	2

```
histogram(~F, data = sim.compF, n = 25)
```



Exploration 9.2: Comparing Popular Diets

```
head(Diets1)
```

Exploration9.2.2

```

  diet BMI
1 Atkins 0.1

```

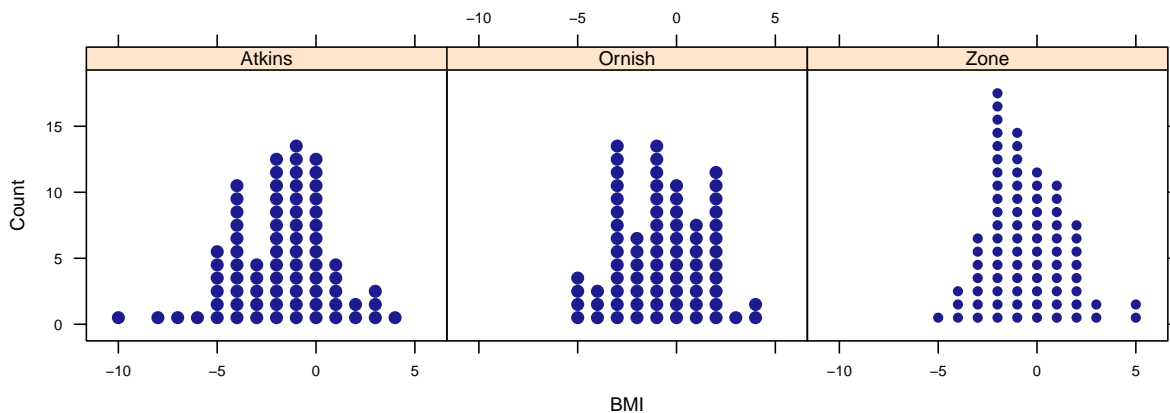
```
2 Atkins -1.0
3 Atkins -5.4
4 Atkins -6.2
5 Atkins -4.1
6 Atkins -1.7
```

Exploration9.2.5

```
favstats(BMI ~ diet, data = Diets1)
```

	diet	min	Q1	median	Q3	max	mean	sd	n	missing
1	Atkins	-9.6	-3.60	-1.50	0.00	4.4	-1.6506494	2.541634	77	0
2	Ornish	-5.1	-2.60	-0.65	0.80	4.3	-0.7697368	2.137788	76	0
3	Zone	-4.6	-1.95	-0.80	1.05	5.1	-0.5303797	2.000920	79	0

```
dotPlot(~BMI | diet, data = Diets1, width = 1)
```



Exploration9.2.6

```
MAD(mean(BMI ~ diet, data = Diets1))
```

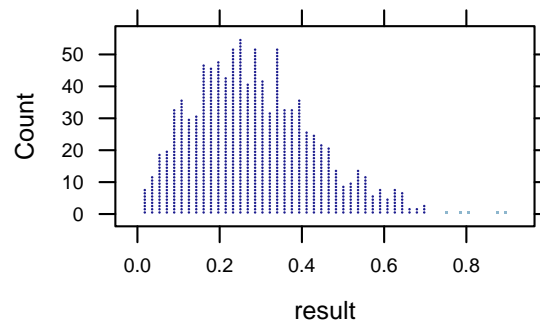
```
[1] 0.7468464
```

```
sim.diet <- do(1000) * MAD(mean(shuffle(BMI) ~ diet, data = Diets1))
head(sim.diet, 3)
```

```
      result
1 0.5271083
2 0.2121364
3 0.2165300
```

```
dotPlot(~result, data = sim.diet, n = 50, groups = (result >= 0.747))
prop(~(result >= 0.747), data = sim.diet)
```

```
TRUE
0.005
```

```
anova(lm(BMI ~ diet, data = Diets1))
```

Exploration9.2.8

Analysis of Variance Table

Response: BMI

```
      Df Sum Sq Mean Sq F value    Pr(>F)
diet      2   53.96  26.9814   5.3916 0.005151 **
Residuals 229 1146.00   5.0044
```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```
sim.dietF <- do(1000) * anova(lm(shuffle(BMI) ~ diet, data = Diets1))
head(sim.dietF, 3)
```

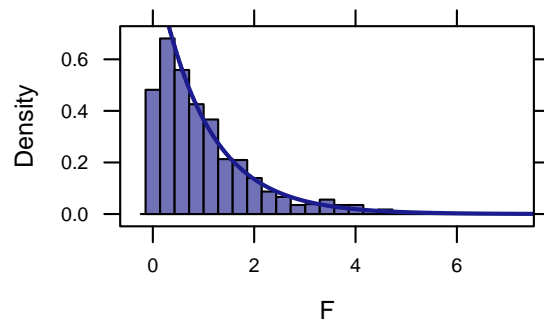
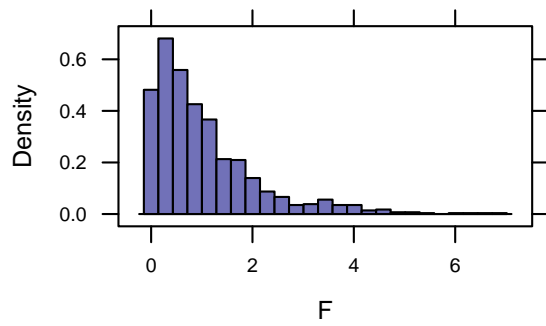
	source	df	SS	MS	F	pval	.row	.index
diet	diet	2	24.27956	12.139781	2.364591	0.09627892	1	1
Residuals	Residuals	229	1175.68315	5.133988	NA	NA	2	1
diet1	diet	2	16.54297	8.271487	1.600591	0.20402629	1	2

```
prop(~(F >= 5.392), data = sim.dietF)
```

```
TRUE
0.005
```

```
histogram(~F, data = sim.dietF, n = 25)
plotDist("f", df1 = 2, df2 = 229, add = TRUE)
```

Exploration9.2.9



```
confint(lm(BMI ~ diet, data = Diets1))
```

Exploration9.2.15

	2.5 %	97.5 %
(Intercept)	-2.1529672	-1.148332
dietOrnish	0.1681949	1.593630
dietZone	0.4143954	1.826144

10

Two Quantitative Variables

10.1 Summarizing the Relationship Between Two Quantitative Variables Using the Correlation Coefficient

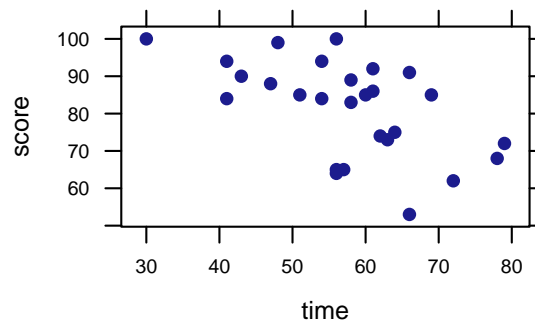
Example 10.1: Exam Times and Exam Scores

Exploring the Data: Graphical Summary

Figure 10.1 plots data that have been modified to exclude 3 observations, so we will take the subset of `ExamTimesScores`.

```
scores <- subset(ExamTimesScores, time < 90)
xyplot(score ~ time, data = scores)
```

Figure10.1



Exploring the Data: Numerical Summary

```
cor(score ~ time, data = scores)
```

Example10.1

```
[1] -0.5636557
```

```
cor(score ~ time, data = ExamTimesScores)
```

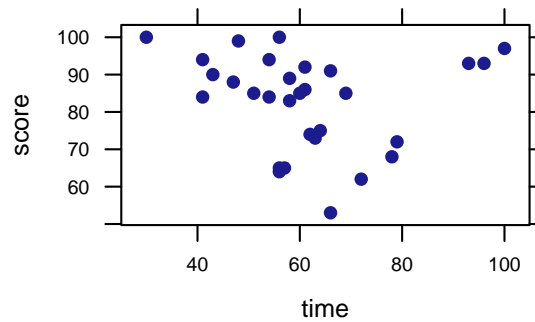
```
[1] -0.124997
```

Caution: Influential Observations

```
xypLOT(score ~ time, data = ExamTimesScores)
cor(score ~ time, data = ExamTimesScores)
```

Figure10.2

```
[1] -0.124997
```



Exploration 10.1: Are Dinner Plates Getting Larger?

```
head(PlateSize)
```

Exploration10.1.2

```
  year  size
1 1950 10.000
2 1956 10.750
3 1957 10.125
4 1958 10.000
5 1963 10.625
6 1964 10.750
```

```
PlateSize
```

Table10.1

```
  year  size
1 1950 10.000
2 1956 10.750
3 1957 10.125
4 1958 10.000
5 1963 10.625
6 1964 10.750
```

```

7  1969  10.625
8  1974  10.000
9  1975  10.500
10 1978  10.125
11 1980  10.375
12 1986  10.750
13 1990  10.375
14 1995  11.000
15 2004  10.750
16 2004  10.125
17 2007  11.500
18 2008  11.000
19 2008  11.125
20 2009  11.000

```

Graphical summary of two-quantitative variables: Scatterplots

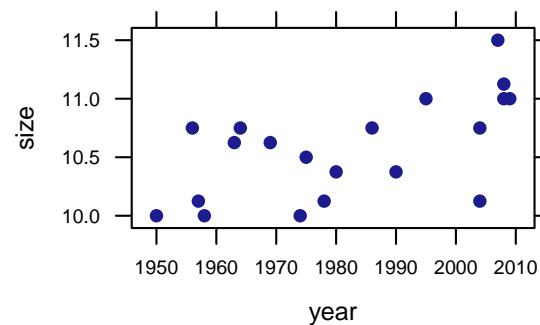
```
cor(size ~ year, data = PlateSize)
```

Exploration10.1.7

```
[1] 0.6037724
```

```
xyplot(size ~ year, data = PlateSize)
```

Exploration10.1.8



Numerical Summaries

```
cor(size ~ year, data = PlateSize)
```

Exploration10.1.15

```
[1] 0.6037724
```

Here is one way to add a new observation to an existing data frame:

Exploration10.1.16

```

PlateSize2 <- PlateSize # make a copy of data with different name
PlateSize2[21, ] <- c(1950, 11.5) # assigning values to the 21st row of data frame
PlateSize2

```

```

  year  size
1  1950 10.000
2  1956 10.750
3  1957 10.125
4  1958 10.000
5  1963 10.625
6  1964 10.750
7  1969 10.625
8  1974 10.000
9  1975 10.500
10 1978 10.125
11 1980 10.375
12 1986 10.750
13 1990 10.375
14 1995 11.000
15 2004 10.750
16 2004 10.125
17 2007 11.500
18 2008 11.000
19 2008 11.125
20 2009 11.000
21 1950 11.500

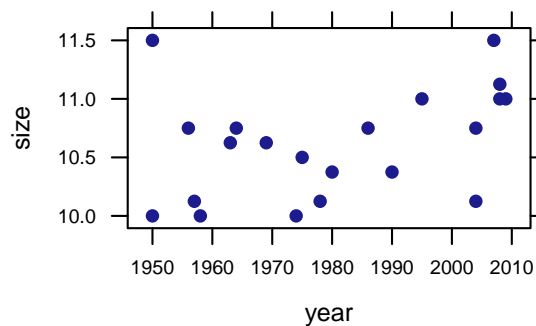
```

```

xyplot(size ~ year, data = PlateSize2)
cor(size ~ year, data = PlateSize2)

```

```
[1] 0.3697467
```



10.2 Inference for the Correlation Coefficient: A Simulation-based Approach

Example 10.2: Exercise Intensity and Mood Changes

ExerciseMood

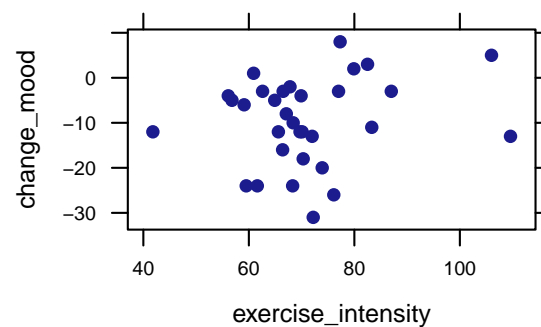
Table10.2

	exercise_intensity	change_mood
1	72.2	-31
2	76.1	-26
3	68.3	-24
4	61.6	-24
5	59.5	-24
6	73.9	-20
7	70.3	-18
8	66.4	-16
9	65.6	-12
10	69.7	-12
11	70.1	-12
12	72.0	-13
13	83.3	-11
14	109.6	-13
15	68.4	-10
16	67.1	-8
17	59.1	-6
18	41.8	-12
19	56.8	-5
20	56.1	-4
21	62.6	-3
22	64.9	-5
23	66.5	-3
24	69.9	-4
25	77.0	-3
26	87.0	-3
27	67.8	-2
28	60.9	1
29	79.9	2
30	82.5	3
31	77.3	8
32	106.0	5

```
xyplot(change_mood ~ exercise_intensity, data = ExerciseMood)
cor(change_mood ~ exercise_intensity, data = ExerciseMood)
```

Figure10.4

```
[1] 0.186898
```



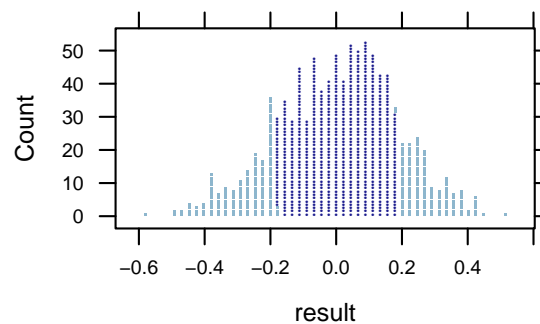
1. $H_0: \rho = 0$
 $H_a: \rho \neq 0$
 Test statistic: $r = 0.187$ (the sample correlation)
2. We simulate a world in which $\rho = 0$:

```
sim.mood <- do(1000) * cor(shuffle(change_mood) ~ exercise_intensity, data = ExerciseMood)
head(sim.mood, 3)
```

result
 1 0.13900806
 2 0.02069769
 3 0.26876050

```
dotPlot(~result, data = sim.mood, n = 50, groups = (result <= -0.187 | result >= 0.187))
```

Figure10.5



3. Strength of evidence:

```
favstats(~result, data = sim.mood)
```

	min	Q1	median	Q3	max	mean	sd	n	missing
	-0.5911367	-0.1202063	0.01280443	0.1277618	0.5046235	0.003412676	0.1805379	1000	0

```
prop(~(result <= -0.187 | result >= 0.187), data = sim.mood)
```

TRUE
 0.297

Figure10.5b

Exploration 10.2: Draft Lottery

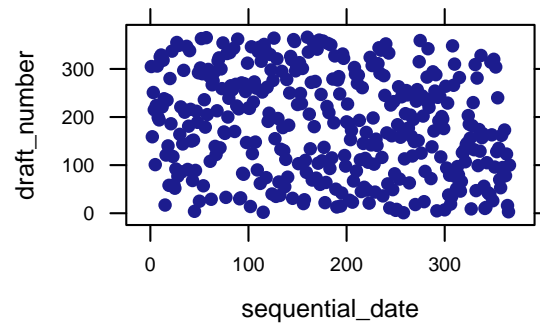
```
head(DraftLottery)
```

	sequential_date	draft_number
1	1	305
2	2	159
3	3	251
4	4	215

Figure10.6


```
5      5      101
6      6      224
```

```
xyplot(draft_number ~ sequential_date, data = DraftLottery)
```



You can identify the specific row in a data set to examine a specific observation like so:

```
DraftLottery[32, ] # draft number for Feb 1
```

Exploration10.2.3

```
  sequential_date draft_number
32              32          86
```

```
median(~draft_number, data = subset(DraftLottery, sequential_date >= 1 & sequential_date <=
  31)) # Jan median
```

Exploration10.2.4

```
[1] 215
```

```
median(~draft_number, data = subset(DraftLottery, sequential_date >= 32 & sequential_date <=
  60)) # Feb median
```

```
[1] 210
```

```
median(~draft_number, data = subset(DraftLottery, sequential_date >= 61 & sequential_date <=
  91)) # Mar median
```

```
[1] 256
```

```
median(~draft_number, data = subset(DraftLottery, sequential_date >= 92 & sequential_date <=
  121)) # Apr median
```

```
[1] 225
```

```
median(~draft_number, data = subset(DraftLottery, sequential_date >= 122 & sequential_date <=
  152)) # May median
```

```
[1] 226

median(~draft_number, data = subset(DraftLottery, sequential_date >= 153 & sequential_date <=
  182)) # Jun median

[1] 207.5

median(~draft_number, data = subset(DraftLottery, sequential_date >= 183 & sequential_date <=
  213)) # Jul median

[1] 188

median(~draft_number, data = subset(DraftLottery, sequential_date >= 214 & sequential_date <=
  243)) # Aug median

[1] 149.5

median(~draft_number, data = subset(DraftLottery, sequential_date >= 244 & sequential_date <=
  274)) # Sep median

[1] 161

median(~draft_number, data = subset(DraftLottery, sequential_date >= 275 & sequential_date <=
  304)) # Oct median

[1] 201.5

median(~draft_number, data = subset(DraftLottery, sequential_date >= 305 & sequential_date <=
  335)) # Nov median

[1] 131

median(~draft_number, data = subset(DraftLottery, sequential_date >= 336 & sequential_date <=
  366)) # Dec median

[1] 100
```

```
cor(draft_number ~ sequential_date, data = DraftLottery)
```

Exploration10.2.5

```
[1] -0.2260414
```

1. $H_0: \rho = 0$

$H_a: \rho \neq 0$

Test statistic: $r = -0.226$ (the sample correlation)

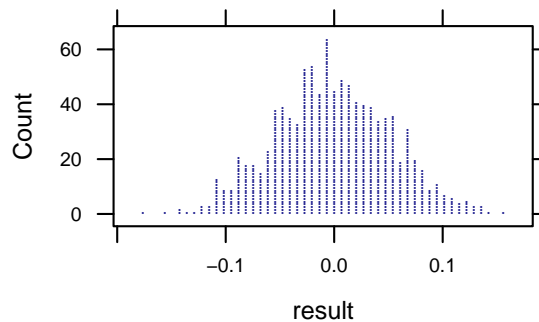
2. We simulate a world in which $\rho = 0$:

```
sim.draft <- do(1000) * cor(shuffle(draft_number) ~ sequential_date, data = DraftLottery)
head(sim.draft, 3)
```

result

```
1 -0.06224926
2  0.02188976
3  0.02020080
```

```
dotPlot(~result, data = sim.draft, n = 50, groups = (result <= -0.226 | result >= 0.226))
```



3. Strength of evidence:

```
favstats(~result, data = sim.draft)
```

	min	Q1	median	Q3	max	mean	sd	n
	-0.1773989	-0.03727075	-0.003127667	0.03597974	0.1547445	-0.001307484	0.05304502	1000
missing	0							

```
prop(~(result <= -0.226 | result >= 0.226), data = sim.draft)
```

```
TRUE
0
```

10.3 Least Squares Regression

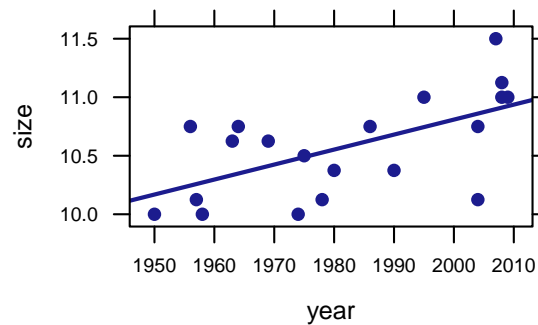
R provides the simple command `lm()` to find the least squares line.

```
xyplot(size ~ year, data = PlateSize, type = c("p", "r"))
lm(size ~ year, data = PlateSize)
```

Call:
`lm(formula = size ~ year, data = PlateSize)`

Coefficients:

(Intercept)	year
-14.8003	0.0128



Note that `type = c("p", "r")` adds the least squares regression line to the scatterplot.

For just the coefficients:

```
coef(lm(size ~ year, data = PlateSize))
```

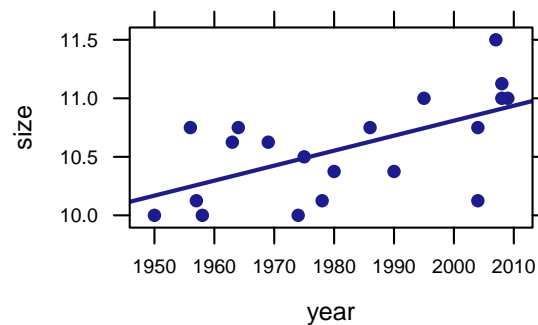
Figure10.7b

(Intercept)	year
-14.80033212	0.01280451

```
xyplot(size ~ year, data = PlateSize, type = c("p", "r"))
resid(lm(size ~ year, data = PlateSize)) # residuals for each point
```

Figure10.8

1	2	3	4	5	6	7
-0.16845690	0.50471606	-0.13308845	-0.27089295	0.29008451	0.40228000	0.21325747
8	9	10	11	12	13	14
-0.47576507	0.01143042	-0.40198310	-0.17759211	0.12058084	-0.30563718	0.25534028
15	16	17	18	19	20	
-0.10990028	-0.73490028	0.60168619	0.08888169	0.21388169	0.07607718	



For more information, including the **coefficient of determination**, use the `summary()` function on the linear model.

```
summary(lm(size ~ year, data = PlateSize))
```

Call:
lm(formula = size ~ year, data = PlateSize)

Residuals:

	Min	1Q	Median	3Q	Max
	-0.73490	-0.20092	0.04375	0.22425	0.60169

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-14.800332	7.897098	-1.874	0.07724 .
year	0.012805	0.003985	3.213	0.00482 **

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3521 on 18 degrees of freedom
Multiple R-squared: 0.3645, Adjusted R-squared: 0.3292
F-statistic: 10.33 on 1 and 18 DF, p-value: 0.004818

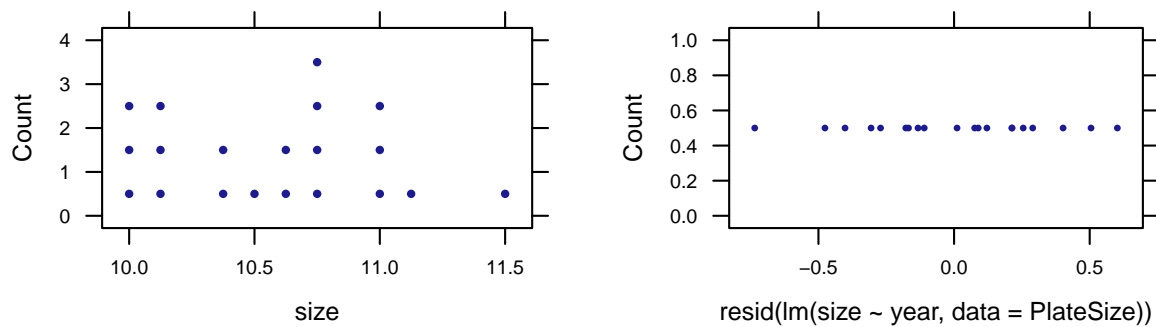
```
rsquared(lm(size ~ year, data = PlateSize)) # just the r-squared
```

```
[1] 0.3645411
```

Figure10.8b

```
dotPlot(~size, data = PlateSize, width = 0.005, cex = 0.25)
dotPlot(~resid(lm(size ~ year, data = PlateSize)), width = 0.001, cex = 0.05)
```

Figure10.9



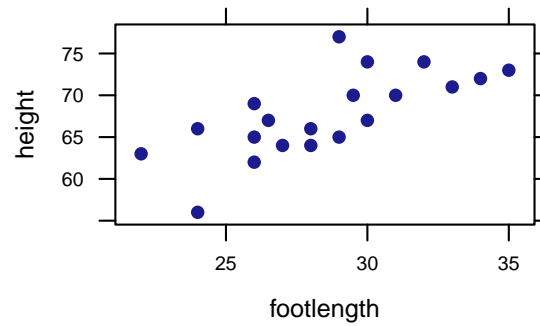
```
head(FootHeight, 3)
```

Exploration10.3.1

```
  footlength height
1       32      74
2       24      66
3       29      77
```

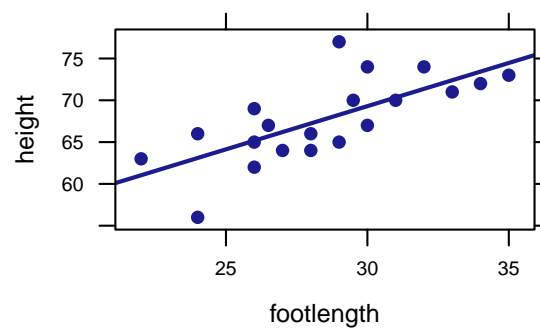
```
xyplot(height ~ footlength, data = FootHeight)
```

Exploration10.3.2



```
xyplot(height ~ footlength, data = FootHeight, type = c("p", "r"))
```

Exploration10.3.4



```
# sum of the absolute values of the residuals of the linear model  
sum(abs(resid(lm(height ~ footlength, data = FootHeight))))
```

Exploration10.3.6

```
[1] 54.59867
```

```
# sum of the squared residuals  
deviance(lm(height ~ footlength, data = FootHeight))
```

Exploration10.3.7

```
[1] 235.0006
```

```
coef(lm(height ~ footlength, data = FootHeight))
```

Exploration10.3.8

```
(Intercept)  footlength  
38.302106    1.033259
```

To make predictions, we can make a function out of the linear model by using the `makeFun()` function.

```
# assigning function of the linear model the name fh
fh <- makeFun(lm(height ~ footlength, data = FootHeight))
fh(footlength = 28) # predicted height for foot length 28
```

Exploration10.3.9

```
1
67.23337
```

```
fh(footlength = 29) # predicted height for foot length 29
```

```
1
68.26663
```

```
fh(footlength = 0) # predicted height for foot length 0
```

Exploration10.3.10

```
1
38.30211
```

```
fh(footlength = 32)
```

Exploration10.3.11

```
1
71.36641
```

```
subset(FootHeight, footlength == "32")
```

```
  footlength height
1          32     74
```

```
subset(FootHeight, footlength == "32")$footlength - fh(footlength = 32)
```

```
1
-39.36641
```

Coefficient of Determination (r^2)

```
cor(height ~ footlength, data = FootHeight)^2
```

Exploration10.3.15

```
[1] 0.5060419
```

```
rsquared(lm(height ~ footlength, data = FootHeight))
```

```
[1] 0.5060419
```

10.4 Inference for Regression Slope: Simulation-Based Approach

Example 10.4: Do students who spend more time in non-academic activities, tend to have lower GPAs?

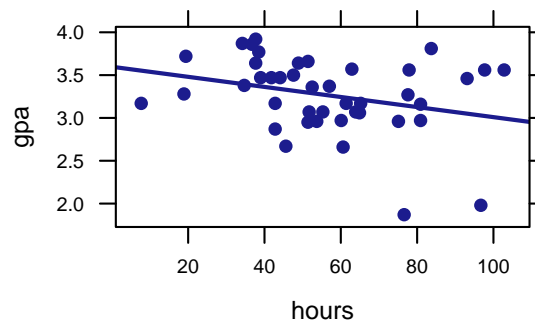
```
xypLOT(gpa ~ hours, data = GPA, type = c("p", "r"))
cor(gpa ~ hours, data = GPA)
```

Figure10.10

```
[1] -0.290021
```

```
coef(lm(gpa ~ hours, data = GPA))
```

```
(Intercept)      hours
3.597690950 -0.005883873
```



1. $H_0: \text{slope} = 0$

$H_a: \text{slope} < 0$

Test statistic: $\text{slope} = -0.00588$ (the sample slope coefficient)

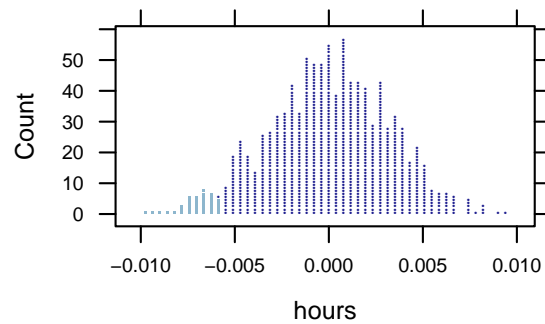
2. We simulate a world in which $\text{slope} = 0$:

```
sim.gpa <- do(1000) * coef(lm(shuffle(gpa) ~ hours, data = GPA))
head(sim.gpa, 3)
```

Figure10.11

```
Intercept      hours
1  3.104523  0.002844396
2  3.037118  0.004037358
3  3.442636 -0.003139649
```

```
dotPlot(~hours, data = sim.gpa, n = 50, groups = (hours <= -0.00588))
```

3. Strength of evidence:

```
favstats(~hours, data = sim.gpa)
```

Figure10.11b

	min	Q1	median	Q3	max	mean
	-0.009634607	-0.002091348	-2.427467e-05	0.002215665	0.009526722	-5.469102e-06
	sd	n	missing			
	0.003180465	1000	0			

```
prop(~(hours <= -0.00588), data = sim.gpa)
```

```
TRUE
0.04
```

Exploration 10.4: Perceptions of Heaviness

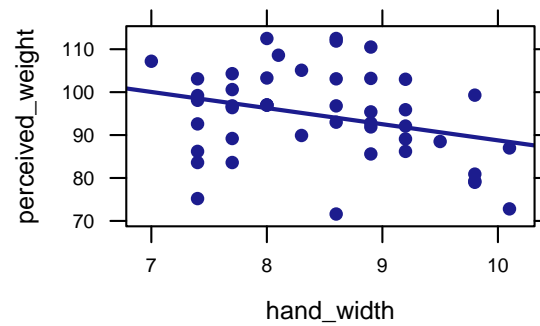
```
head(Handwidth, 10)
```

Table10.4

	hand_width	perceived_weight
1	7.4	75.2
2	7.4	83.6
3	7.4	86.2
4	7.4	92.6
5	7.4	98.1
6	7.4	99.2
7	7.4	103.1
8	7.0	107.2
9	7.7	104.3
10	7.7	100.6

```
xyplot(perceived_weight ~ hand_width, data = Handwidth, type = c("p", "r"))
```

Exploration10.4.2



```
coef(lm(perceived_weight ~ hand_width, data = Handwidth))
```

Exploration10.4.3

```
(Intercept)  hand_width
126.333411   -3.756255
```

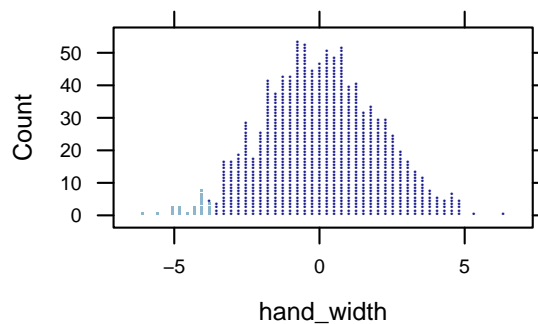
1. $H_0: \text{slope} = 0$
 $H_a: \text{slope} < 0$
 Test statistic: $\text{slope} = -3.756$ (the sample slope coefficient)
2. We simulate a world in which $\text{slope} = 0$:

```
sim.hand <- do(1000) * coef(lm(shuffle(perceived_weight) ~ hand_width, data = Handwidth))
head(sim.hand, 3)
```

Exploration10.4.5

```
Intercept  hand_width
1  84.58420  1.17306710
2  94.14856  0.04380425
3 110.28287 -1.86117008
```

```
dotPlot(~hand_width, data = sim.hand, n = 50, groups = (hand_width <= -3.756))
```



3. Strength of evidence:

```
favstats(~hand_width, data = sim.hand)
```

Exploration10.4.6

	min	Q1	median	Q3	max	mean	sd	n	missing
	-6.075354	-1.276914	0.0005374755	1.385924	6.331703	0.05301637	1.953232	1000	0

```
prop(~(hand_width <= -6.756), data = sim.hand)
```

```
TRUE
0
```

10.5 Inference for the Regression Slope: Theory-Based Approach

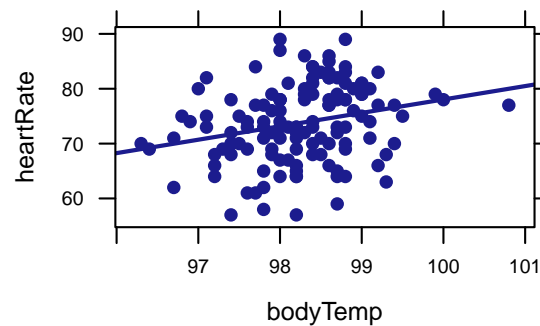
Example 10.5A: Predicting Heart Rate from Body Temperature

```
head(TempHeart)
```

Figure10.13

	bodyTemp	heartRate
1	96.3	70
2	96.7	71
3	96.9	74
4	97.0	80
5	97.1	73
6	97.1	75

```
xyplot(heartRate ~ bodyTemp, data = TempHeart, type = c("p", "r"))
```



```
coef(lm(heartRate ~ bodyTemp, data = TempHeart))
```

Figure10.14

	bodyTemp
(Intercept)	-166.284719
	2.443238

1. $H_0: \text{slope} = 0$

H_a : slope $\neq 0$

Test statistic: slope = 2.443 (the sample slope coefficient)

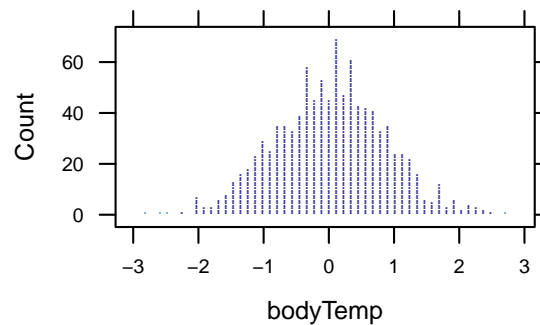
2. We simulate a world in which slope = 0:

```
sim.rate <- do(1000) * coef(lm(shuffle(heartRate) ~ bodyTemp, data = TempHeart))
head(sim.rate, 3)
```

	Intercept	bodyTemp
1	207.04719	-1.3566075
2	140.17331	-0.6759521
3	-92.32675	1.6904793

```
dotPlot(~bodyTemp, data = sim.rate, n = 50, groups = (bodyTemp <= -2.443 | bodyTemp >= 2.443))
```

Figure10.14b



3. Strength of evidence:

```
favstats(~bodyTemp, data = sim.rate)
```

	min	Q1	median	Q3	max	mean	sd	n	missing
	-2.847704	-0.5331875	0.06166499	0.5945086	2.678295	0.02867916	0.8458917	1000	0

```
prop(~(bodyTemp <= -2.443 | bodyTemp >= 2.443), data = sim.rate)
```

```
TRUE
0.004
```

Figure10.14c

```
sim.ratet <- do(1000) * coef(summary(lm(shuffle(heartRate) ~ bodyTemp, data = TempHeart)))
head(sim.ratet, 10)
```

Figure10.15

	Estimate	Std..Error	t.value	Pr.> t
(Intercept)	43.2628666	83.6046573	0.517469576	0.6057215
bodyTemp	0.3104215	0.8509211	0.364806406	0.7158580
(Intercept).1	15.2098429	83.4878494	0.182180317	0.8557294
bodyTemp.1	0.5959507	0.8497323	0.701339342	0.4843644
(Intercept).2	93.1349087	83.6305785	1.113646591	0.2675177
bodyTemp.2	-0.1971860	0.8511850	-0.231660546	0.8171716

```
(Intercept).3 -0.2334883 83.3920153 -0.002799888 0.9977704
bodyTemp.3    0.7531359 0.8487569 0.887340005 0.3765601
(Intercept).4 98.8021862 83.6188202 1.181578332 0.2395637
bodyTemp.4    -0.2548686 0.8510653 -0.299470137 0.7650671
```

```
coef(summary(lm(heartRate ~ bodyTemp, data = TempHeart)))
```

Figure10.16

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-166.284719	80.912346	-2.055122	0.041901345
bodyTemp	2.443238	0.823519	2.966826	0.003591489

```
confint(lm(heartRate ~ bodyTemp, data = TempHeart))
```

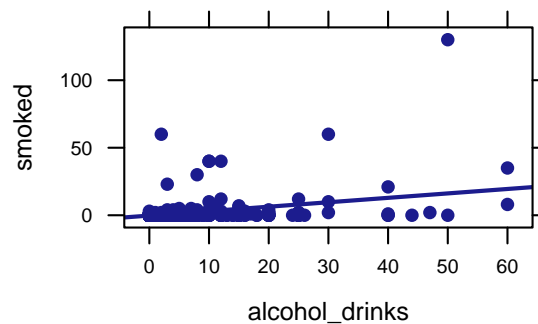
Figure10.17

	2.5 %	97.5 %
(Intercept)	-326.383620	-6.185819
bodyTemp	0.813765	4.072711

Example 10.5B: Smoking and Drinking

```
xyplot(smoked ~ alcohol_drinks, data = AlcoholSmoke, type = c("p", "r"))
```

Figure10.18



Caution: Outliers and Influential Observations

```
cor(smoked ~ alcohol_drinks, data = AlcoholSmoke)
```

Example10.5B

```
[1] 0.3703078
```

```
cor(smoked ~ alcohol_drinks, data = subset(AlcoholSmoke, smoked < 125))
```

```
[1] 0.3014187
```

Exploration 10.5: Predicting Brain Density from Number of Facebook Friends

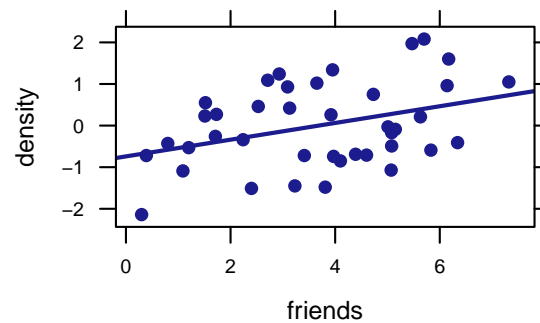
```
head(Facebook)
```

Table10.5

	friends	density
1	0.30	-2.14
2	1.09	-1.09
3	0.39	-0.72
4	1.20	-0.53
5	0.80	-0.43
6	1.71	-0.26

```
xypplot(density ~ friends, data = Facebook, type = c("p", "r"))
```

Exploration10.5.2



```
coef(lm(density ~ friends, data = Facebook))
```

Exploration10.5.3

```
(Intercept)    friends
-0.7404404    0.2008952
```

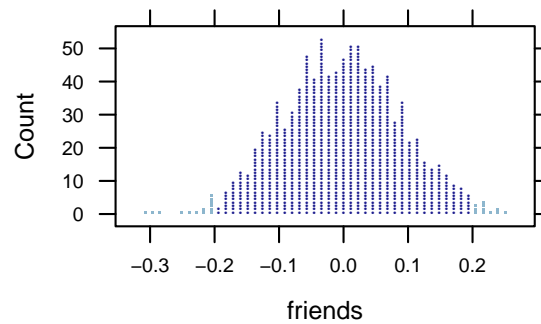
1. $H_0: \text{slope} = 0$
 $H_a: \text{slope} \neq 0$
 Test statistic: $\text{slope} = 0.201$ (the sample slope coefficient)
2. We simulate a world in which $\text{slope} = 0$:

```
sim.face <- do(1000) * coef(lm(shuffle(density) ~ friends, data = Facebook))
head(sim.face, 3)
```

Exploration10.5.4

```
Intercept    friends
1  0.14535911 -0.04008954
2 -0.28238865  0.07628066
3  0.04101383 -0.01170206
```

```
dotPlot(~friends, data = sim.face, n = 50, groups = (friends <= -0.201 | friends >= 0.201))
```



3. Strength of evidence:

```
favstats(~friends, data = sim.face)
```

Exploration10.5.4b

	min	Q1	median	Q3	max	mean	sd	n
	-0.3038165	-0.06430713	-0.001944393	0.05924783	0.2549344	-0.00280994	0.09102442	1000
missing	0							

```
prop(~(friends <= -0.201 | friends >= 0.201), data = sim.face)
```

```
TRUE
0.025
```

```
cor(density ~ friends, data = Facebook)
```

Exploration10.5.6

```
[1] 0.3655156
```

```
coef(summary(lm(density ~ friends, data = Facebook)))
```

Exploration10.5.11

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-0.7404404	0.33947697	-2.181121	0.03543210
friends	0.2008952	0.08299091	2.420689	0.02037788

```
summary(lm(density ~ friends, data = Facebook))
```

Exploration10.5.12

```
Call:
lm(formula = density ~ friends, data = Facebook)
```

```
Residuals:
```

Min	1Q	Median	3Q	Max
-----	----	--------	----	-----

```
-1.5050 -0.8057 -0.0401  0.6734  1.6753
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-0.74044	0.33948	-2.181	0.0354 *
friends	0.20090	0.08299	2.421	0.0204 *

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9421 on 38 degrees of freedom

Multiple R-squared: 0.1336, Adjusted R-squared: 0.1108

F-statistic: 5.86 on 1 and 38 DF, p-value: 0.02038

```
confint(lm(density ~ friends, data = Facebook))
```

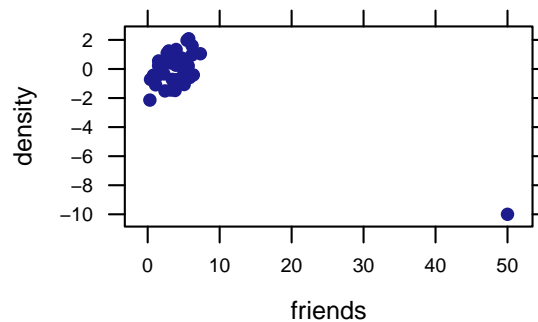
Exploration10.5.15

	2.5 %	97.5 %
(Intercept)	-1.42767560	-0.05320521
friends	0.03288886	0.36890148

```
Facebook2 <- Facebook # make a copy of data with different name
Facebook2[41, ] <- c(50, -10) # assigning values to the 41st row of data frame
xyplot(density ~ friends, data = Facebook2)
cor(density ~ friends, data = Facebook2)
```

Exploration10.5.16

```
[1] -0.7735351
```



```
summary(lm(density ~ friends, data = Facebook2))
```

Exploration10.5.16b

Call:

```
lm(formula = density ~ friends, data = Facebook2)
```

Residuals:

Min	1Q	Median	3Q	Max
-2.75772	-0.74164	-0.07364	0.84703	2.49728

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	0.67522	0.22108	3.054	0.00405	**
friends	-0.19167	0.02515	-7.622	3.04e-09	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.185 on 39 degrees of freedom

Multiple R-squared: 0.5984, Adjusted R-squared: 0.5881

F-statistic: 58.1 on 1 and 39 DF, p-value: 3.042e-09

ExamTimesScores., 313
 FirstBase2, 273
 GameSims., 16
 ISIwithR, 5, 7
 NightLight1, 298
 bargraph(), 9
 bargraph, 10
 binom.test(), 46, 81
 binom.test, 88, 93
 bwplot(), 9, 64
 data(), 6, 7
 densityplot(), 9
 do(), 16
 dotPlot(), 12
 do, 16
 favestats, 14
 freqpolygon(), 18
 histogram(), 9, 38
 lattice, 14
 lm(), 321
 makeFun(), 325
 mosaicplot, 10
 mosaic, 5, 7, 15
 mosiac, 10
 pnorm(), 40, 41
 pnorm, 39
 prop(), 23
 prop.test(), 41, 46, 81
 qnorm, 39
 rflip(), 15
 rflip, 16
 rnorm, 64
 round(), 64
 sample(), 50
 sim.sci, 23
 summary(), 8
 summary, 323
 tally(), 8
 tally, 9
 xpnorm(), 28, 30
 xpnorm, 28, 29, 39