An R Companion to Introduction to Statistical Investigations
Preliminary Edition

Randall Pruim and Lana Park

August 30, 2014

Contents

0 Preliminaries

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:

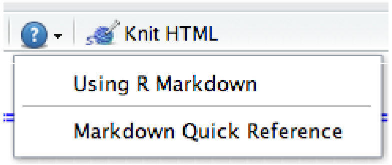
- `mosaic` (a package from Project MOSAIC)
- `Tintle1` (data sets)

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.

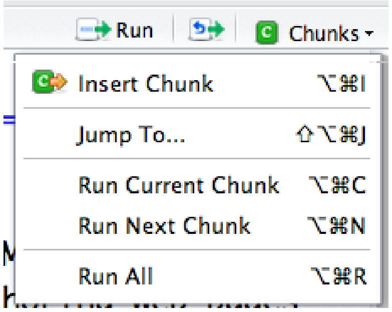
To create a new R Markdown document, go to "File", "New", then "R Markdown."

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 "Mardown 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 displayng it, (b) run R code without displaying the output, (c) controling 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:

function-syntax

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

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:

$$\boxed{}\,(\,\boxed{} \sim \boxed{}\,,\ \texttt{data} = \boxed{}\,)$$

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

$$\boxed{\texttt{goal}}\,(\,\boxed{\texttt{y}} \sim \boxed{\texttt{x}}\,,\ \texttt{data} = \boxed{\texttt{mydata}}\,)$$

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 funcitonality 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 Tintle1 which comes with a number of data sets.

```
require(Tintle1)  # 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
 opt-in :55   donor:108
 opt-out:50   not  : 53
 neutral:56
```

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


'data.frame': 161 obs. of  2 variables:
 $ default: Factor w/ 3 levels "opt-in","opt-out",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ choice : Factor w/ 2 levels "donor","not": 1 1 1 1 1 1 1 1 1 1 ...


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)



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

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


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


       default
choice   opt-in opt-out neutral Total
  donor      23      41      44   108
  not        32       9      12    53
  Total      55      50      56   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  opt-in opt-out neutral
  donor 0.4182  0.8200  0.7857
  not   0.5818  0.1800  0.2143


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


       default
choice  opt-in opt-out neutral
  donor  41.82   82.00   78.57
  not    58.18   18.00   21.43
```

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`).

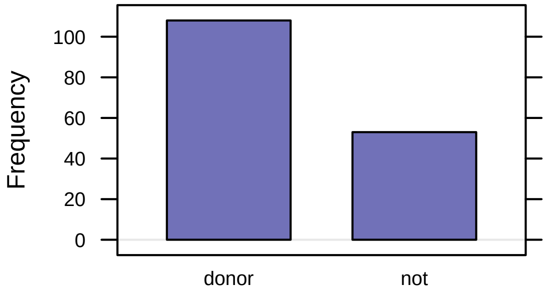
R also has many tools to visualize data.The general syntax for making a graph of one variable in a data frame is

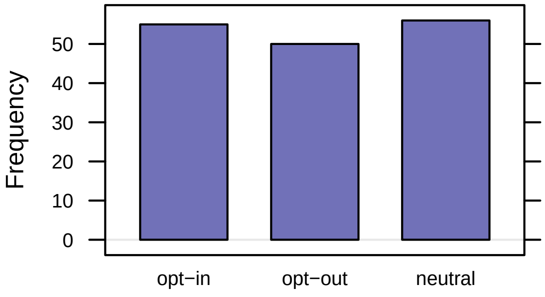
```
plotname(~variable, data = dataName)
```

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)
```

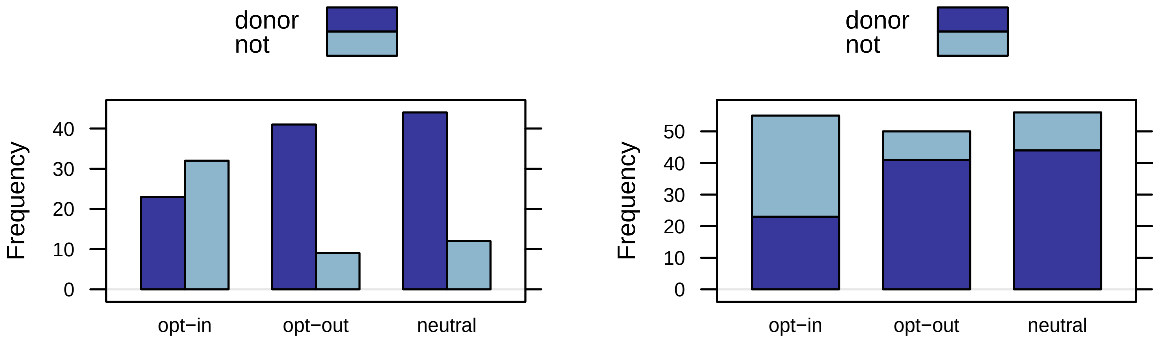




Notice that the `bargraph()` uses the frequency, or counts.

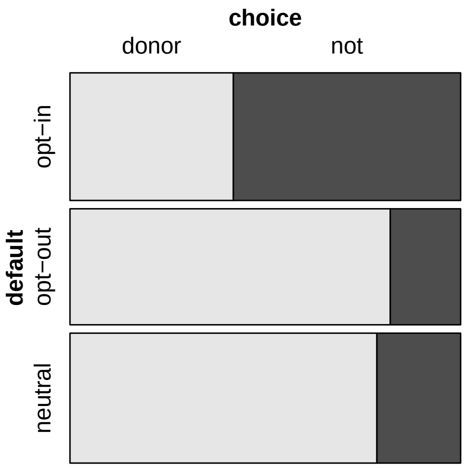
In order to graph the variable `default` and show what `choice` each option made, we can utilize the argument `groups=`.

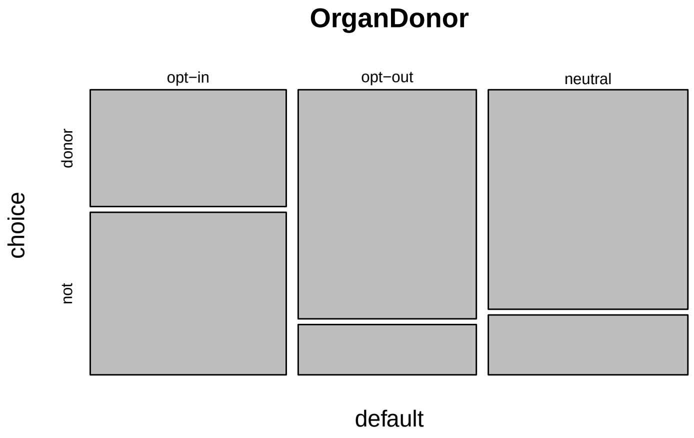
```
bargraph(~default, groups = choice, data = OrganDonor, auto.key = TRUE)
bargraph(~default, groups = choice, data = OrganDonor, auto.key = TRUE, stack = TRUE)
```



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

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





0.2 Exploring Data

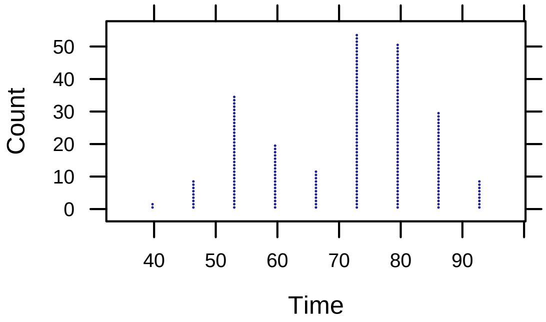
Example P.2: Old Faithful

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
 1st Qu.:60
 Median :75
 Mean   :71
 3rd Qu.:81
 Max.   :95
```

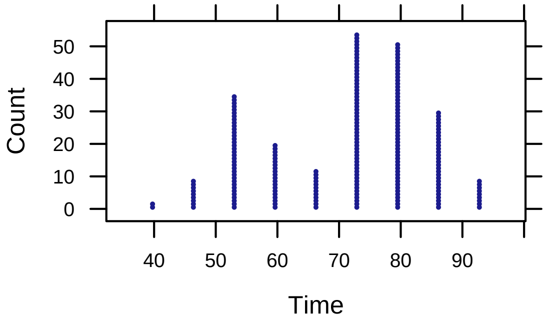
Another useful graph for examining the **shape**, **center**, and **variability** is the **dotplot**:

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



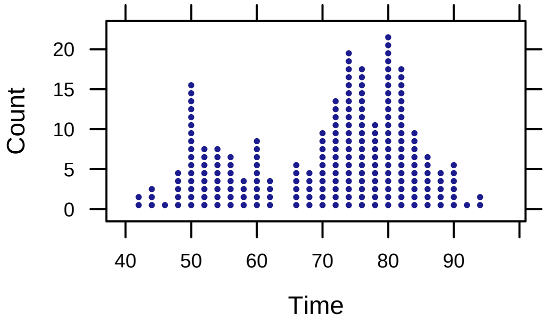
The dots in this plot are a bit small. The defaults for `dotPlot()` may not be the best way to examine a particular data set. We can increase the size of the dots using the `cex` argument. (`cex` stands for "character expansion" and is used to scale up or down the size of plotting characters – in this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

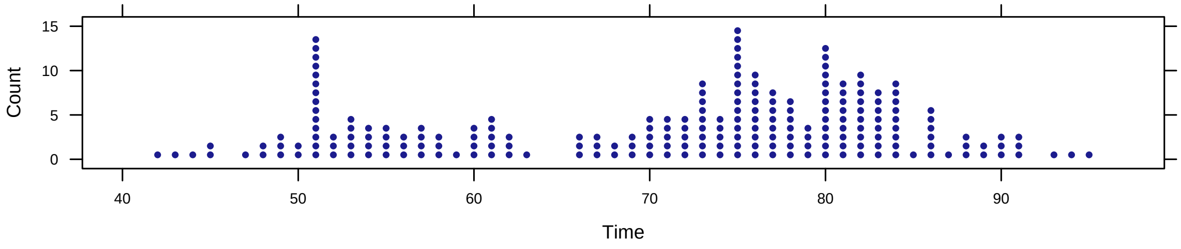


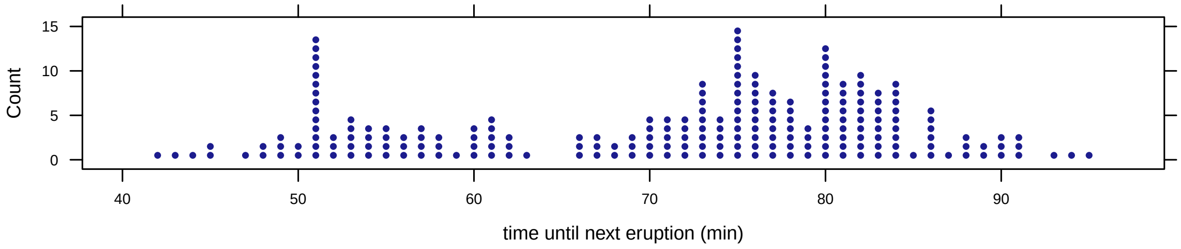
Notice that the dots have been automatically resized when we do this.

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.

FigureP.3

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

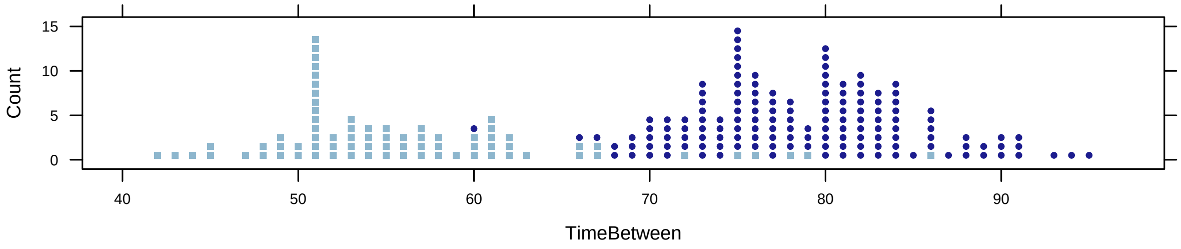




Similar to the bargraph, we can organize the variables a little differently for the dotplot to graph them in relation to one another.

```
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
 long :146    Min.   :42
 short: 76    1st Qu.:60
              Median :75
              Mean   :71
              3rd Qu.:81
              Max.   :95
```

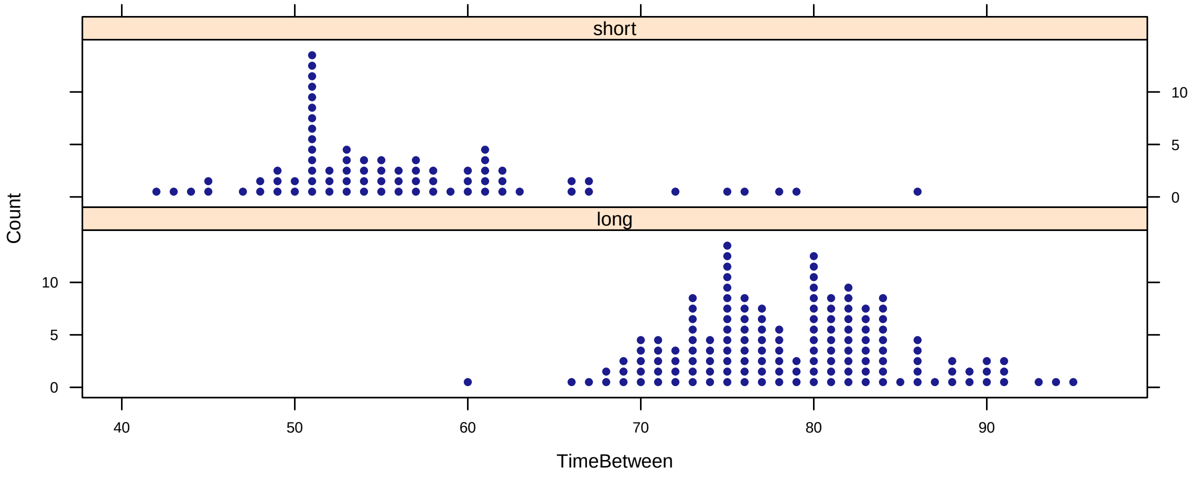
```
dotPlot(~TimeBetween, groups = EruptionType, data = OldFaithful2, width = 1)
```



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 `favestats()` for "favorite" statistics.

TableP.1

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

```
 min Q1 median Q3 max  mean   sd   n missing
  42 60     75 81  95 71.01 12.8 222       0
```

```
favstats(TimeBetween ~ EruptionType, data = OldFaithful2)
```

```
  .group min Q1 median    Q3 max  mean    sd   n missing
1   long  60 75   78.5 83.00  95 78.69 6.252 146       0
2  short  42 51   54.0 60.25  86 56.25 8.457  76       0
```

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

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


[1] 71.01


sd(~TimeBetween, data = OldFaithful2)


[1] 12.8
```

```
mean(TimeBetween ~ EruptionType, data = OldFaithful2)


 long short
78.69 56.25


sd(TimeBetween ~ EruptionType, data = OldFaithful2)


 long short
6.252 8.457
```

```
mean(~TimeBetween | EruptionType, data = OldFaithful2)


 long short
78.69 56.25


sd(~TimeBetween | EruptionType, data = OldFaithful2)


 long short
6.252 8.457
```

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 ] ...

T T H H T

Number of Heads: 2 [Proportion Heads: 0.4]
```

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) = 0.333333333333333 ] ...

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

Number of Heads: 6 [Proportion Heads: 0.4]
```

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)
```

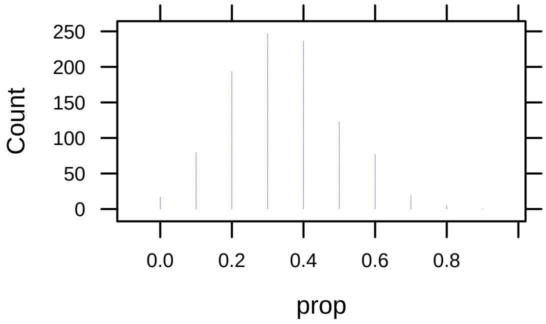
```
Loading required package:  parallel
```

```
head(GameSims)
```

```
   n heads tails prop
1 10     7     3  0.7
2 10     4     6  0.4
3 10     6     4  0.6
4 10     3     7  0.3
5 10     2     8  0.2
6 10     3     7  0.3
```

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)
```

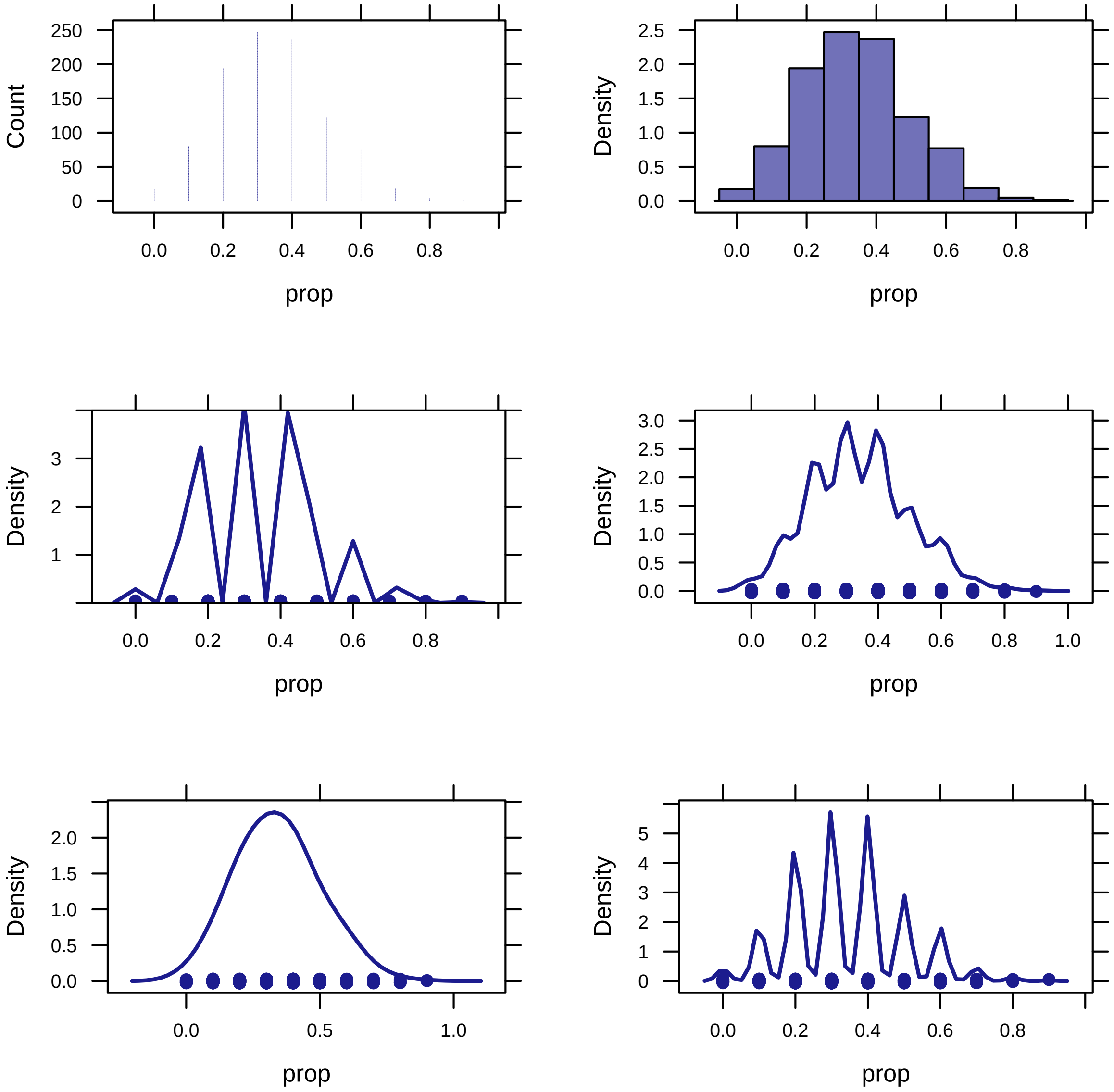


0.4 Other Visualizations

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 in 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.

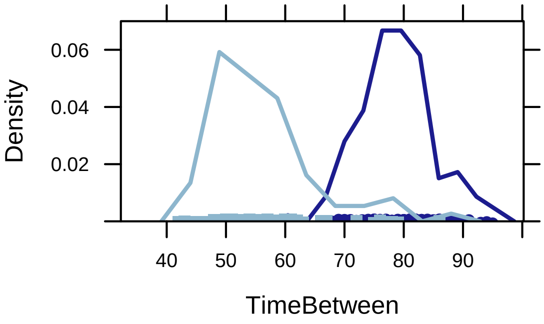
```
    dotPlot(˜ prop, data = GameSims, width = 0.1)
  histogram(˜ prop, data = GameSims, width = 0.1)
freqpolygon(˜ prop, data = GameSims, width = 0.1, ylim=c(0,4))
densityplot(˜ prop, data = GameSims)
densityplot(˜ prop, data = GameSims, adjust=2)    # "smoother"
densityplot(˜ prop, data = GameSims, 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 0.3416 0.1544 1000       0

       mean(˜ prop, data = GameSims)

[1] 0.3416

         sd(˜ prop, data = GameSims)

[1] 0.1544
```

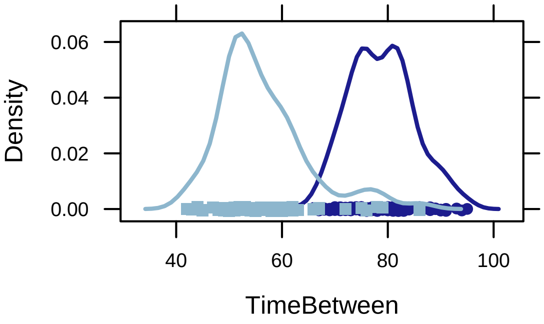


For this data set, a histogram is probably best. This is in part due to the discreteness of the data – there are only 11 possible values for `prop`.

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

Figure1.2

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


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

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

Number of Heads: 6 [Proportion Heads: 0.375]
```

Figure1.3

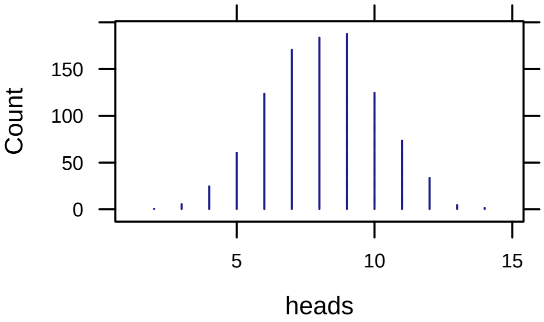
```
rflip(n = 16, prob = 0.5)  # another sequence of 16 coin flips


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

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

Number of Heads: 9 [Proportion Heads: 0.5625]
```

Using and evaluating the coin flip chance model

Figure1.4

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

Loading required package:  parallel
```

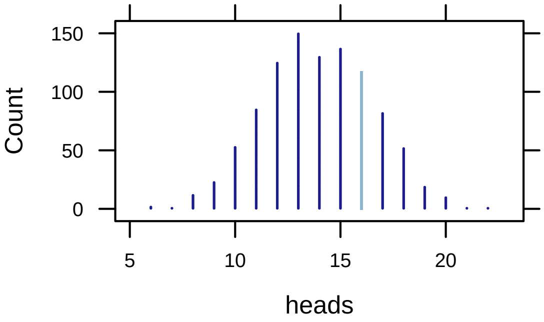
```
head(sim, 3)


   n heads tails   prop
1 16     4    12 0.2500
2 16     5    11 0.3125
3 16    11     5 0.6875


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



Another Doris and Buzz study

Figure1.6

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


Loading required package:  parallel


head(sim2, 3)


   n heads tails   prop
1 28    18    10 0.6429
2 28     9    19 0.3214
3 28    13    15 0.4643


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

Exploration1.1.13

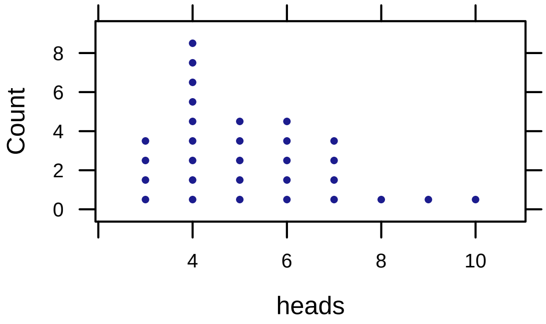
```
sim.harley <- do(1) * rflip(10, 0.5)
sim.harley
```

```
   n heads tails prop
1 10     8     2  0.8
```

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

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

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

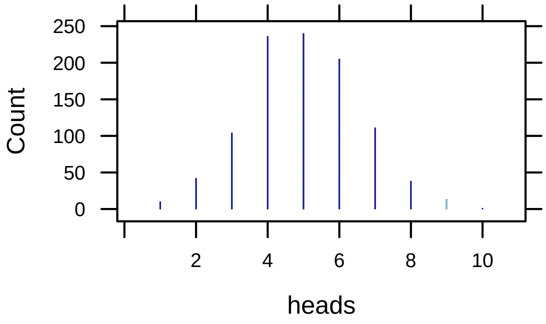


Exploration1.1.14

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

```
   n heads tails prop
1 10     3     7  0.3
2 10     6     4  0.6
3 10     4     6  0.4
```

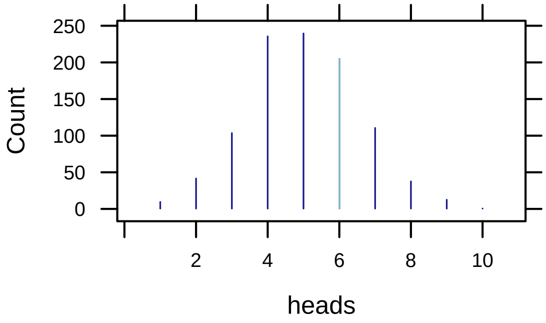
```
dotPlot(~heads, data = sim.harley2, width = 1, cex = 3, groups = (heads == 9))
```



Another Study

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

Exploration1.1.23



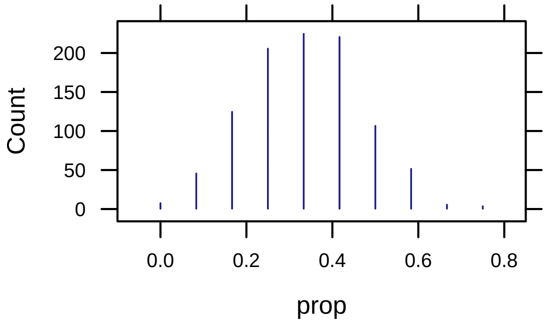
1.2 Measuring the Strength of Evidence

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 $\pi = 1/3$:

Figure1.7

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

   n heads tails   prop
1 12     7     5 0.5833
2 12     4     8 0.3333
3 12     3     9 0.2500

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

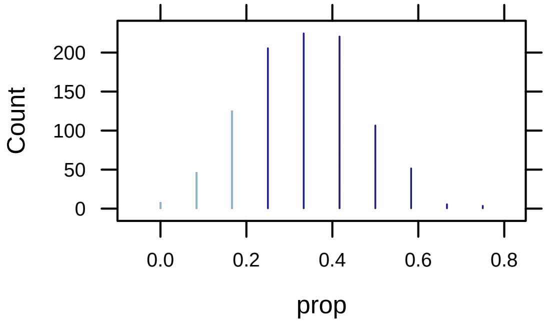


3. Strength of evidence:

   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`.

Figure1.8

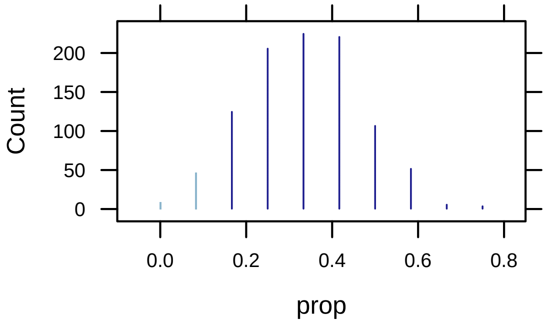
```
dotPlot(~prop, data = sim.sci, cex = 3, width = 1/12, groups = (prop <= 1/6))
prop(~(prop <= 1/6), data = sim.sci)

 TRUE
0.179
```



Conclusions

Figure1.9

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

 TRUE
0.054
```



Exploration 1.2: Tasting Water

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

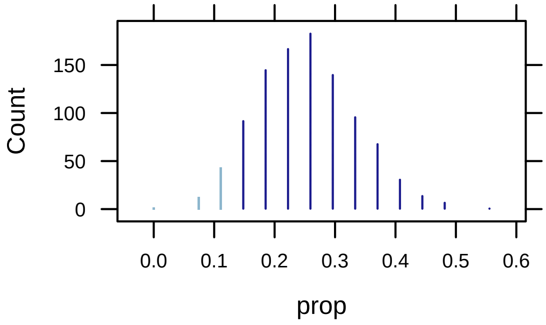
   H_a: $\pi < 1/4$

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

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

Exploration1.2.18

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

   n heads tails   prop
1 27     6    21 0.2222

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

   n heads tails   prop
1 27     4    23 0.1481
2 27     5    22 0.1852
3 27    10    17 0.3704

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



3. Strength of evidence:

Exploration1.2.20

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

 TRUE
0.056
```

Alternate Analysis

1. H_0: $\pi = 3/4$

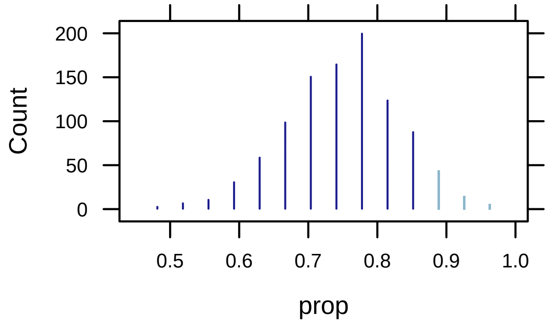
   H_a: $\pi > 3/4$

   Test statistic: $\hat{p} = 0.889$ (the sample proportion of 24/27)

2. We simulate a world in which $\pi = 3/4$:

Exploration1.2.26

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

   n heads tails   prop
1 27    21     6 0.7778
2 27    22     5 0.8148
3 27    20     7 0.7407

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



3. Strength of evidence:

Exploration1.2.26b

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

 TRUE
0.062
```

1.3 Alternative Measure of Strength of Evidence

Example 1.3: Heart Transplant Operations

1. H_0: $\pi = 0.15$

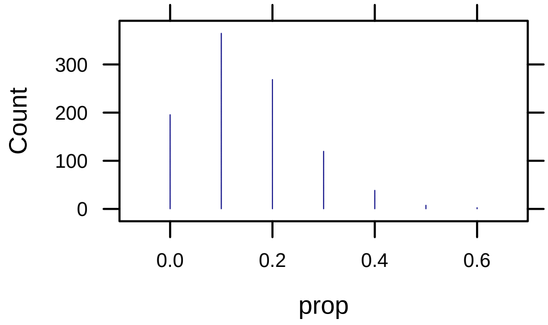
H_a: $\pi > 0.15$

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

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

Figure1.10

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

   n heads tails prop
1 10     0    10  0.0
2 10     1     9  0.1
3 10     2     8  0.2

mean(~prop, data = sim.heart)

[1] 0.1477

sd(~prop, data = sim.heart)

[1] 0.1128

favstats(~prop, data = sim.heart)

 min  Q1 median  Q3 max   mean     sd    n missing
   0 0.1    0.1 0.2 0.6 0.1477 0.1128 1000       0

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



3. Strength of evidence:

Figure1.10b

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

TRUE
   0
```

Digging deeper into the St. George's mortality data

1. H_0: $\pi = 0.15$

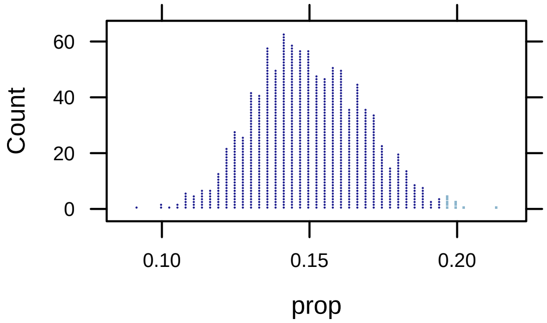
H_a: $\pi > 0.15$

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

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

Figure1.11

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

    n heads tails   prop
1 361    39   322 0.1080
2 361    44   317 0.1219
3 361    64   297 0.1773

favstats(~prop, data = sim.1986)

     min     Q1 median     Q3    max   mean      sd    n missing
 0.09141 0.1357 0.1496 0.1634 0.2133 0.1498 0.01851 1000       0

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



3. Strength of evidence:

Figure1.11b

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

TRUE
0.01
```

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:

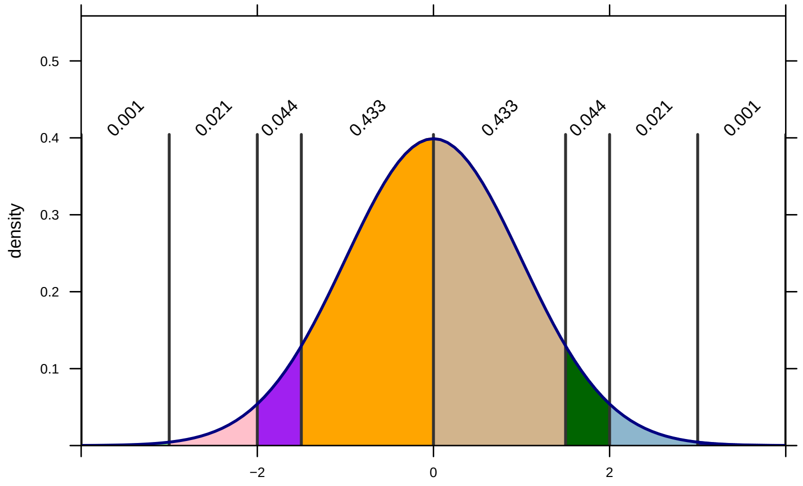
Example1.3

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

[1] 2.593

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

[1] 5.752
```

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.

Figure1.12

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


If X ~ 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] 0.00135 0.02275 0.06681 0.50000 0.93319 0.97725 0.99865
```



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.

Example1.3b

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


If X ~ 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] 0.9952

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



If X ~ N(0.15,0.113), then

P(X <= 0.8) = P(Z <= 5.752) = 1
P(X >  0.8) = P(Z >  5.752) = 0
[1] 1
```

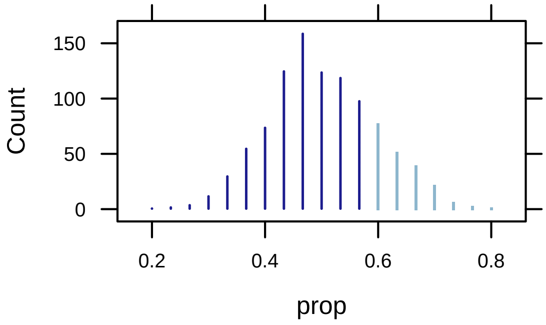
We'll ignore the p-values and plots for now and just realize that `xpnorm()` has computed the z-score for us so that we do not need to manually compute z by using R as a calculator.

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 $\pi = 0.5$:

Exploration1.3.7

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

   n heads tails   prop
1 30    16    14 0.5333
2 30    13    17 0.4333
3 30    14    16 0.4667

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



3. Strength of evidence:

Exploration1.3.7b

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

 TRUE
0.197
```

Exploration1.3.8

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

[1] 0.5004

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

[1] 0.0946

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

[1] 1.057
```

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

Figure1.13

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


If X ~ N(0.5,0.094598037975366), then

P(X <= 0.6) = P(Z <= 1.057) = 0.8548
P(X >  0.6) = P(Z >  1.057) = 0.1452
[1] 0.8548
```

1.4 What Impacts Strength of Evidence?

Example 1.4: Predicting Elections from Faces?

1. H_0: $\pi = 0.5$

   H_a: $\pi > 0.5$

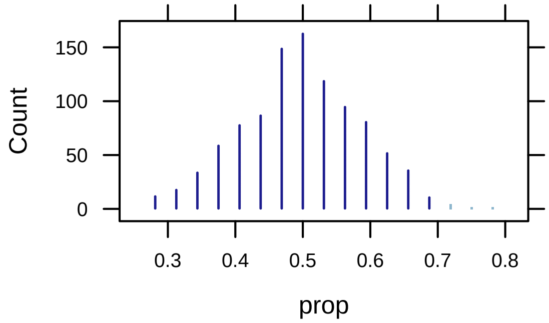
   Test statistic: $\hat{p} = 0.719$ (the sample proportion of 23/32)

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

   Figure1.14

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

      n heads tails   prop
   1 32    19    13 0.5938
   2 32    15    17 0.4688
   3 32    15    17 0.4688

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

```
    min     Q1 median     Q3    max   mean      sd    n missing
 0.2812 0.4375    0.5 0.5625 0.7812 0.4968 0.08796 1000       0

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



3. Strength of evidence:

Figure1.14b

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

 TRUE
0.006
```

Strength of evidence with the standardized statistic:

Figure1.14c

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

[1] 0.4968

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

[1] 0.08796

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


If X ~ N(0.5,0.0879625447806297), then

P(X <= 0.71875) = P(Z <= 2.487) = 0.9936
P(X >  0.71875) = P(Z >  2.487) = 0.0064
[1] 0.9936
```

What impacts strength of evidence?

Figure1.15

```
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)
```

```
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)
```

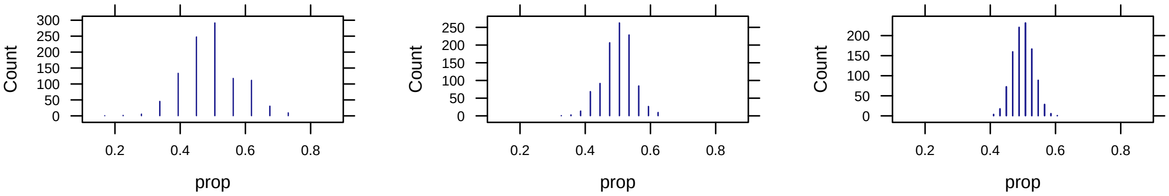


Figure1.15b

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

[1] 0.08524

sd(~prop, data = senate.128)

[1] 0.04626

sd(~prop, data = senate.256)

[1] 0.03136
```

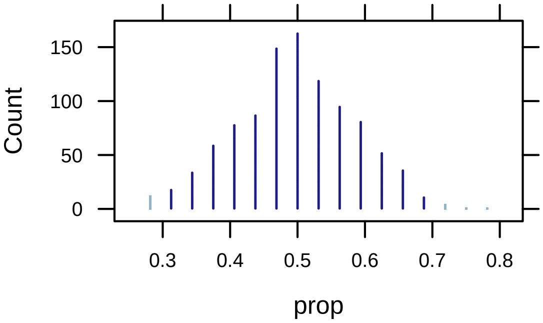
Figure1.15c

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

TRUE
0.01

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

TRUE
   0

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

TRUE
   0
```

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 $\pi = 0.5$:

Figure1.16

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



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:

Figure1.16b

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

 TRUE
0.018
```

Follow-up Study

1. H_0: $\pi = 0.5$

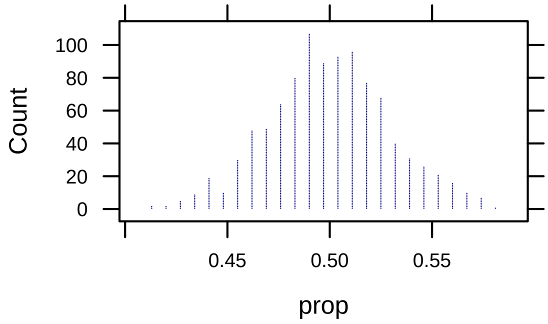
   H_a: $\pi \neq 0.5$

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

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

Figure1.17

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

    n heads tails   prop
1 279   144   135 0.5161
2 279   153   126 0.5484
3 279   138   141 0.4946

favstats(~prop, data = sim.house)

    min     Q1 median     Q3    max   mean      sd    n missing
 0.4122 0.4803 0.4982 0.5197 0.5842 0.4994 0.02986 1000       0

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



3. Strength of evidence:

Figure1.17b

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

TRUE
   0
```

Strength of evidence with the standardized statistic:

Figure1.17c

```
mean(˜prop, data = sim.house)

[1] 0.4994

sd <- sd(˜prop, data = sim.house)
sd

[1] 0.02986

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


If X ˜ N(0.5,0.0298637127792574), then

P(X <= 0.67741935483871) = P(Z <= 5.941) = 1
P(X >  0.67741935483871) = P(Z >  5.941) = 0
[1] 1
```

Exploration 1.4: Competitive Advantage to Uniform Colors?

1. H_0: $\pi = 0.5$

   H_a: $\pi > 0.5$

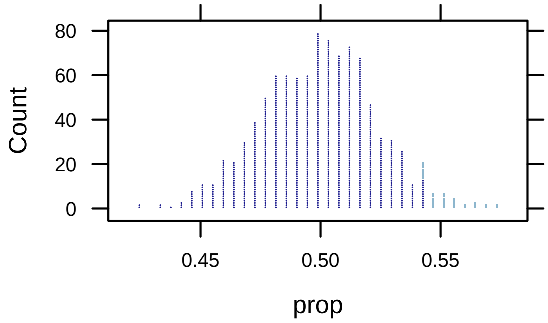
   Test statistic: $\hat{p} = 0.543$ (the sample proportion of 248/457)

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

Exploration1.4.3

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

```
    n heads tails   prop
1 457   230   227 0.5033
2 457   232   225 0.5077
3 457   237   220 0.5186

favstats(~prop, data = sim.red)

    min     Q1 median     Q3    max   mean      sd    n missing
 0.4245 0.4836 0.5011 0.5164 0.5733 0.5005 0.02387 1000       0

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



3. Strength of evidence:

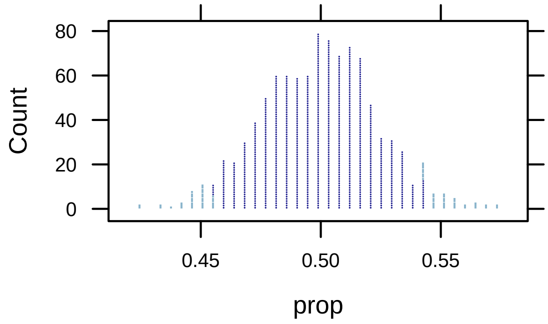
Exploration1.4.3b

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

 TRUE
0.036
```

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:

Exploration1.4.5

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



3. Strength of evidence:

Exploration1.4.5b

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

 TRUE
0.069
```

Difference between statistic and null hypothesis parameter value

1. H_0: $\pi = 0.5$

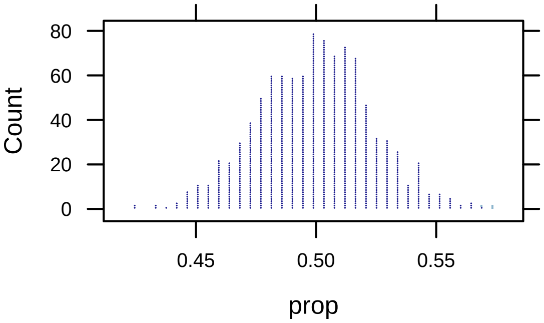
   H_a: $\pi > 0.5$

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

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

Exploration1.4.6

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



3. Strength of evidence:

Exploration1.4.6b

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

 TRUE
0.003
```

Sample size

1. H_0: $\pi = 0.5$

   H_a: $\pi > 0.5$

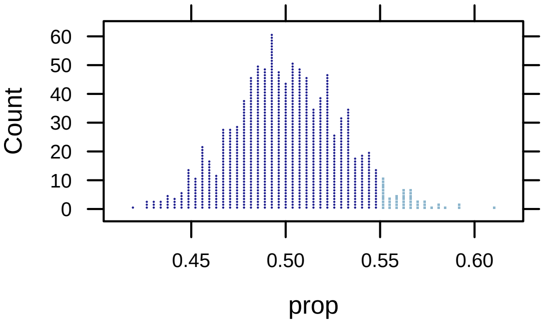
   Test statistic: $\hat{p} = 0.551$ (the sample proportion of 150/272)

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

Exploration1.4.7

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

```
    n heads tails   prop
1 272   157   115 0.5772
2 272   138   134 0.5074
3 272   143   129 0.5257

favstats(~prop, data = sim.box)

    min     Q1 median     Q3    max   mean      sd    n missing
 0.4191 0.4816    0.5 0.5221 0.6103 0.5011 0.02906 1000       0

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



3. Strength of evidence

Exploration1.4.7b

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

 TRUE
0.047
```

1.5 Inference on a single proportion: Theory-based approach

Example 1.5: Halloween Treats

1. H_0: $\pi = 0.5$

   H_a: $\pi \neq 0.5$

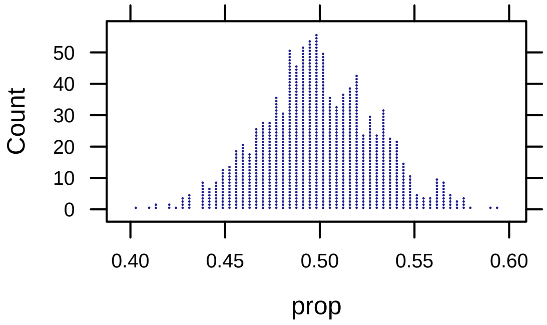
   Test statistic: $\hat{p} = 0.523$ (the sample proportion of 148/283)

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

Figure1.19

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

    n heads tails   prop
1 283   137   146 0.4841
2 283   157   126 0.5548
3 283   132   151 0.4664
```

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

    min     Q1 median     Q3    max   mean      sd    n missing
 0.4028 0.4806 0.4982 0.5194 0.5936 0.4991 0.03041 1000       0

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



Theory-based approach (One proportion z test)

Calculating predicted standard deviation:

Example1.5

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

[1] 0.02972
```

Calculating z-score:

Example1.5b

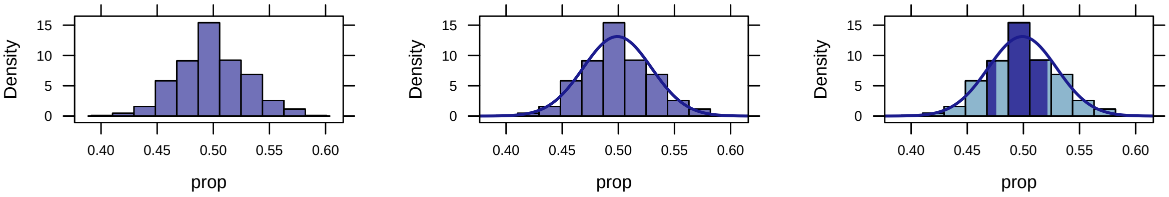
```
z <- (0.523 - mean)/sd
z

[1] 0.7738

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


If X ~ N(0.5,0.0297219149138882), then

P(X <= 0.522968197879859) = P(Z <= 0.773) = 0.7802
P(X >  0.522968197879859) = P(Z >  0.773) = 0.2198
[1] 0.7802
```

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

Figure1.20

```
histogram(~prop, data = sim.candy)
histogram(~prop, data = sim.candy, fit = "normal")
histogram(~prop, data = sim.candy, fit = "normal", group = (prop <= 135/283 | prop >= 148/283))
prop(~(prop <= 135/283 | prop >= 148/283), data = sim.candy)
```

```
 TRUE
0.472
```

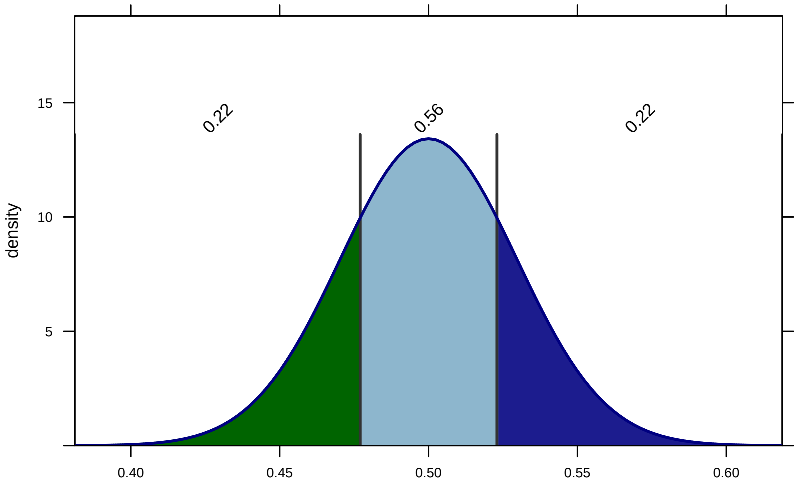


Now that we've covered normal approximation, we can examine the rest of the output from `xpnorm()`. Because it's a two-sided test, we can input both the observed statistic (148/283) and the statistic that is as extreme as the observed (135/283).

Figure1.20b

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

```
If X ~ 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] 0.2198 0.7802
```



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($Z <= -0.773$) 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`.

Figure1.20c

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



If X ~ N(0.5,0.0297219149138882), then

P(X <= 0.522968197879859) = P(Z <= 0.773) = 0.7802
P(X >  0.522968197879859) = P(Z >  0.773) = 0.2198
[1] 0.2198


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



If X ~ N(0.5,0.0297219149138882), then

P(X <= 0.522968197879859) = P(Z <= 0.773) = 0.7802
P(X >  0.522968197879859) = P(Z >  0.773) = 0.2198
[1] 0.4397
```

This results in the p-value of the alternative hypothsis that π is greater than the observed statistic (the default is the alternative hypothsis 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:

Figure1.20d

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


[1] 0.4397
```

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

Figure1.20e

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


[1] 0.439
```

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

Example1.5c

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



1-sample proportions test with continuity correction
```

```
data:  x and n
X-squared = 0.5088, df = 1, p-value = 0.4756
alternative hypothesis: true p is not equal to 0.5
95 percent confidence interval:
 0.4631 0.5822
sample estimates:
    p
0.523
```

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.

Figure1.5d

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


1-sample proportions test without continuity correction

data:  x and n
X-squared = 0.5972, df = 1, p-value = 0.4397
alternative hypothesis: true p is not equal to 0.5
95 percent confidence interval:
 0.4649 0.5805
sample estimates:
    p
0.523
```

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

Example1.5e

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


[1] 0.1361
```

Figure1.21

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


 TRUE
0.179
```

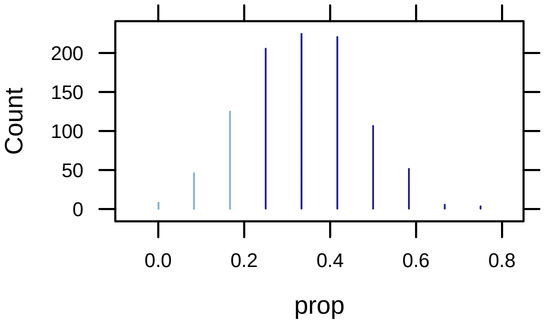
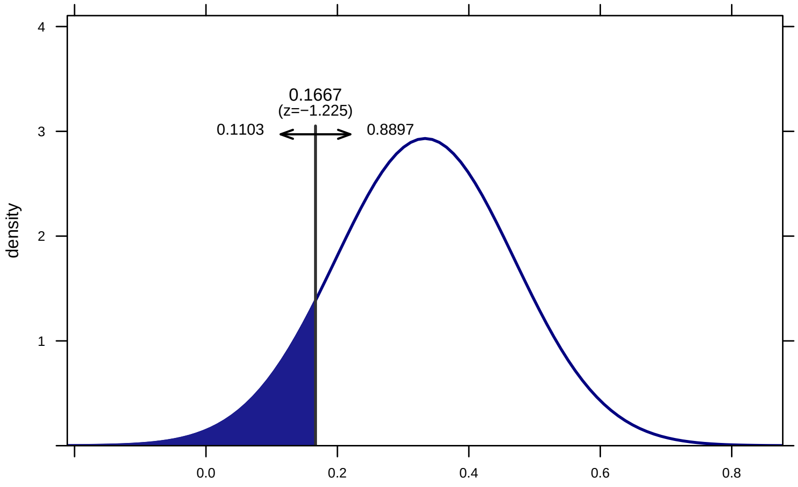


Figure1.21b

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

```
If X ~ N(0.333333333333333,0.136082763487954), then

P(X <= 0.166666666666667) = P(Z <= -1.225) = 0.1103
P(X >  0.166666666666667) = P(Z >  -1.225) = 0.8897
[1] 0.1103
```



Exploration 1.5: Calling Heads or Tails

1. H_0: $\pi = 0.5$

   H_a: $\pi > 0.5$

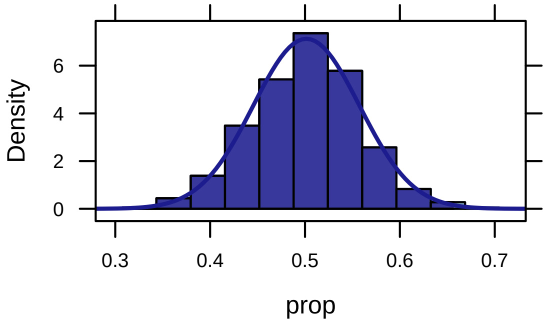
   Test statistic: $\hat{p} = 0.651$ (the sample proportion of 54/83)

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

Exploration1.5.5

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

```
   n heads tails   prop
1 83    46    37 0.5542
2 83    43    40 0.5181
3 83    41    42 0.4940

favstats(˜prop, data = sim.heads)

    min     Q1 median     Q3    max   mean      sd    n missing
 0.3253 0.4699  0.506 0.5422 0.6867 0.5014 0.05601 1000       0

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



3. Strength of evidence

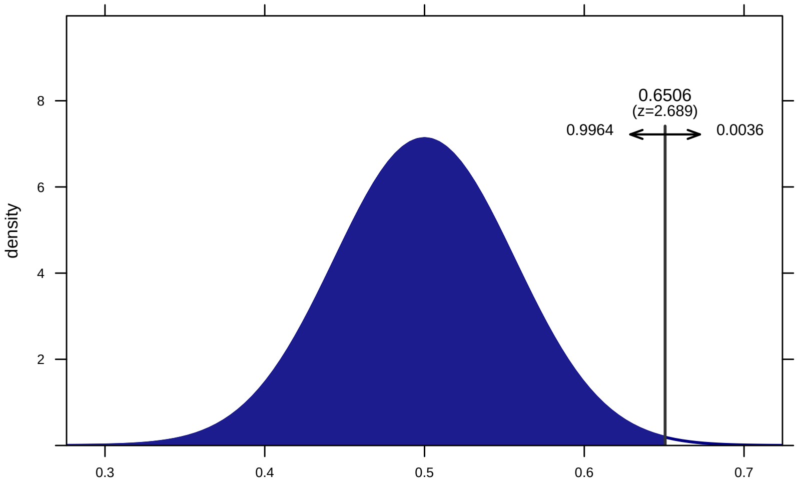
Exploration1.5.5b

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

 TRUE
0.005
```

Normal approximation using simulated sd:

Exploration1.5.5c

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


If X ˜ N(0.5,0.0560149681052275), then

P(X <= 0.650602409638554) = P(Z <= 2.689) = 0.9964
P(X >  0.650602409638554) = P(Z >  2.689) = 0.0036
[1] 0.003588
```



Formulas

Exploration1.5.8

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


[1] 0.05488
```

Exploration1.5.9

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



If X ~ N(0.5,0.0548821299948452), then

P(X <= 0.650602409638554) = P(Z <= 2.744) = 0.997
P(X >  0.650602409638554) = P(Z >  2.744) = 0.003
[1] 0.003034


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



1-sample proportions test without continuity correction

data:  x and n
X-squared = 7.53, df = 1, p-value = 0.003034
alternative hypothesis: true p is greater than 0.5
95 percent confidence interval:
 0.561 1.000
sample estimates:
     p
0.6506
```

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:

Exploration1.5.12

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

[1] 0.05488

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


If X ~ N(0.5,0.0548821299948452), then

P(X <= 0.7) = P(Z <= 3.644) = 0.9999
P(X >  0.7) = P(Z >  3.644) = 1e-04
[1] 0.0002683
```

Approximate test for proportions without continuity correction:

Exploration1.5.12b

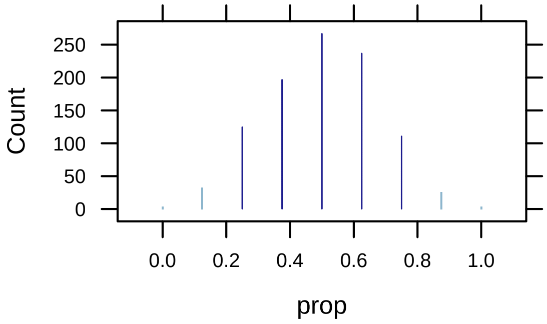
```
prop.test(58.1, 83, correct = FALSE)  # 58.1 = 0.70 * 83


1-sample proportions test without continuity correction

data:  x and n
X-squared = 13.28, df = 1, p-value = 0.0002683
alternative hypothesis: true p is not equal to 0.5
95 percent confidence interval:
 0.5944 0.7879
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 $\pi = 0.5$:

Exploration1.5.13

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

  n heads tails prop
1 8     4     4 0.50
2 8     4     4 0.50
3 8     2     6 0.25

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



3. Strength of evidence:

Exploration1.5.13b

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

 TRUE
0.063
```

Approximate test for proportions without continuity correction:

Exploration1.5.13c

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


1-sample proportions test without continuity correction

data:  x and n
X-squared = 4.5, df = 1, p-value = 0.03389
alternative hypothesis: true p is not equal to 0.5
95 percent confidence interval:
 0.5291 0.9776
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

data:  x and n
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:
 0.4735 0.9968
sample estimates:
probability of success
                 0.875
```

```
binom.test(58, 83)



Exact binomial test

data:  x and n
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:
 0.5882 0.7947
sample estimates:
probability of success
                0.6988
```

2

Generalization: How Broadly Do the Results Apply?

2.1 Sampling from a Finite Population

Example 2.1A: Sampling Students

Table2.1

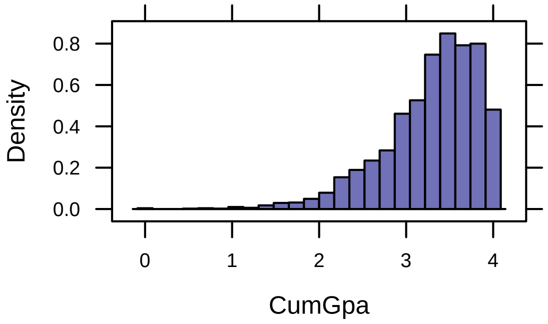
```
head(CollegeMidwest, 8)
```

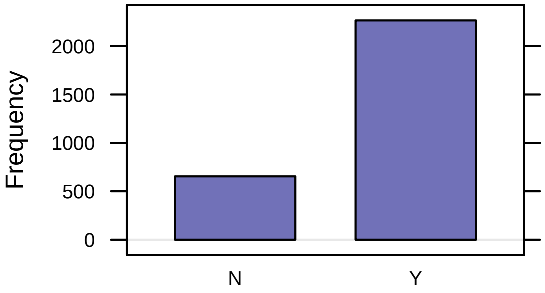
```
  OnCampus CumGpa
1        N   2.92
2        N   3.59
3        N   3.36
4        N   2.47
5        N   3.46
6        Y   2.98
7        Y   3.07
8        Y   3.79
```

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 `nint,` or `n` for short.

Figure2.1

```
histogram(~CumGpa, data = CollegeMidwest, n = 24)
bargraph(~OnCampus, data = CollegeMidwest)
```





Simple Random Samples

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

Table2.2

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


     OnCampus CumGpa orig.ids
1054        Y   3.90     1054
940         Y   3.40      940
1828        Y   3.33     1828
1668        Y   3.85     1668
2161        Y   3.76     2161
2637        Y   2.91     2637
1364        Y   3.91     1364
818         Y   2.66      818
1233        Y   3.91     1233
1817        N   3.69     1817
1147        Y   3.59     1147
398         Y   3.51      398
2495        Y   3.54     2495
2516        Y   3.05     2516
1486        N   3.74     1486
1837        Y   2.58     1837
1798        Y   3.35     1798
2571        Y   2.86     2571
2099        Y   3.51     2099
1980        Y   3.23     1980
698         Y   4.00      698
616         Y   2.36      616
70          N   3.58       70
1313        Y   3.25     1313
1952        Y   2.12     1952
1345        Y   3.95     1345
1503        N   3.39     1503
2115        Y   3.98     2115
2652        Y   2.76     2652
783         N   3.71      783


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

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

Table2.3

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

[1] 3.379

mean(~CumGpa, data = sample2)

[1] 3.379

mean(~CumGpa, data = sample3)

[1] 3.318

mean(~CumGpa, data = sample4)

[1] 3.262

mean(~CumGpa, data = sample5)

[1] 3.112

prop(~OnCampus, level = "Y", data = sample1)

     Y
0.8333

prop(~OnCampus, level = "Y", data = sample2)

     Y
0.7667

prop(~OnCampus, level = "Y", data = sample3)

     Y
0.6667

prop(~OnCampus, level = "Y", data = sample4)

  Y
0.8

prop(~OnCampus, level = "Y", data = sample5)

  Y
0.8
```

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.

Figure2.2

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

```
Loading required package:  parallel
```

```
head(sample.gpa)
```

```
  result
1  3.212
2  3.269
3  3.382
4  3.087
5  3.268
6  3.239
```

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

```
   min    Q1 median    Q3   max  mean      sd    n missing
 2.965 3.233    3.3 3.366 3.571 3.295 0.09986 1000       0
```

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

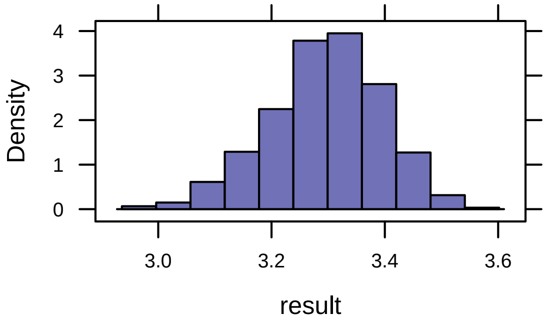
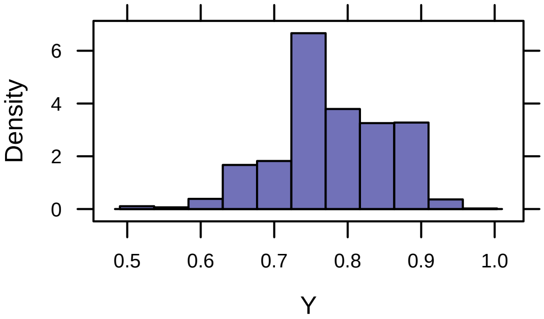


Figure2.2b

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

```
       Y
1 0.6667
2 0.7667
3 0.7667
4 0.8333
5 0.7333
6 0.8000
```

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


 min     Q1 median     Q3    max   mean      sd    n missing
 0.5 0.7333 0.7833 0.8333 0.9667 0.7795 0.07462 1000       0


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



Exploration 2.1A: Sampling Words

```
head(GettysburgAddress)


[1] "Four"  "score" "and"   "seven" "years" "ago"


words <- sample(GettysburgAddress, 10)
nchar(words[1:10])


 [1]  6  5  7  7  3  6 10  6  4  3
```

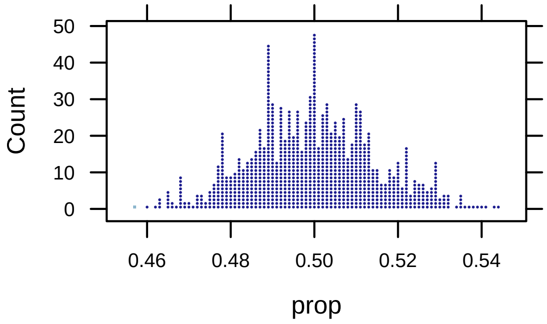
Example 2.1B: Should Supersize Drinks be Banned?

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 $\pi = 0.5$:

Figure2.3

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


     n heads tails   prop
1 1093   542   551 0.4959
2 1093   553   540 0.5059
3 1093   506   587 0.4629


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

```
    min     Q1 median     Q3    max   mean      sd    n missing
 0.4575 0.4895 0.4995 0.5096 0.5444 0.4997 0.01502 1000       0

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



3. Strength of evidence:

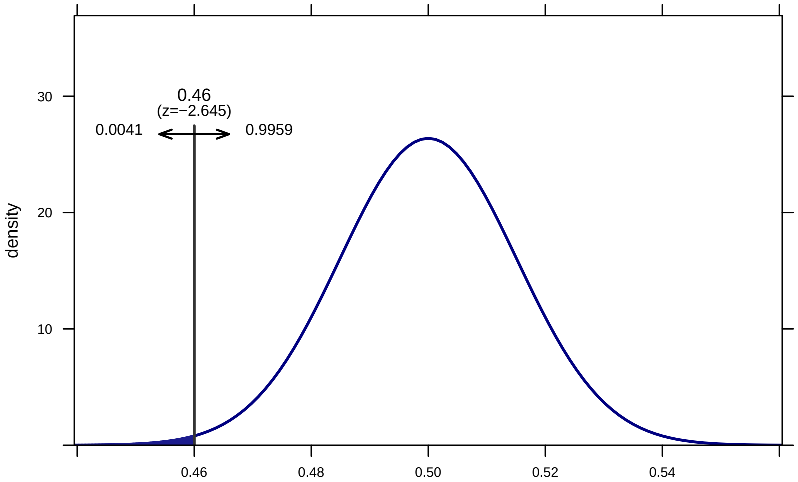
Figure2.3b

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

 TRUE
0.001
```

Normal approximation using predicted standard deviation:

Figure2.4

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

[1] 0.01512

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] 0.004086
```



Approximate test for proportions with continuity correction:

Figure2.4b

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


1-sample proportions test with continuity correction

data:  x and n
X-squared = 6.767, df = 1, p-value = 0.004644
alternative hypothesis: true p is less than 0.5
95 percent confidence interval:
 0.0000 0.4855
sample estimates:
     p
0.4602
```

Exact test for proportions:

Figure2.4c

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


Exact binomial test

data:  x and n
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:
 0.0000 0.4855
sample estimates:
probability of success
                0.4602
```

Exploration 2.1B: Banning Smoking in Cars?

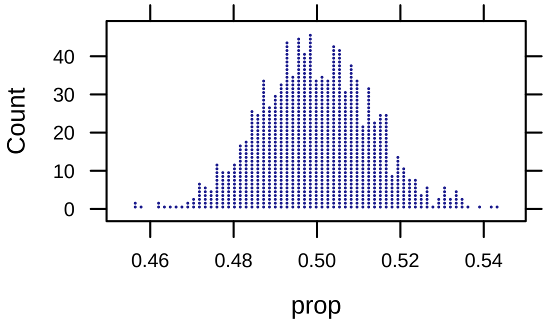
1. H_0: $\pi = 0.5$
   H_a: $\pi > 0.5$

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

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

Exploration2.1B.10

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

     n heads tails   prop
1 1421   730   691 0.5137
2 1421   670   751 0.4715
3 1421   695   726 0.4891

favstats(˜prop, data = sim.smoke)

   min     Q1 median     Q3    max   mean      sd    n missing
 0.456 0.4905 0.4996 0.5088 0.5426 0.4999 0.01359 1000       0

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



3. Strength of evidence:

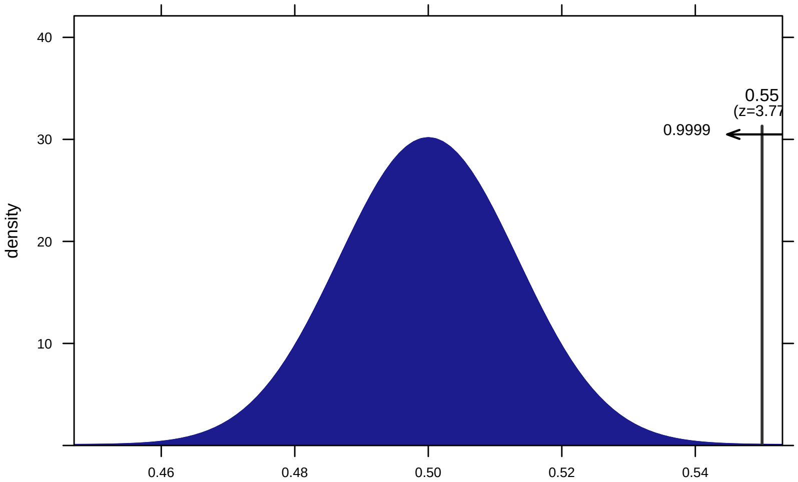
Exploration2.1B.10b

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

TRUE
   0
```

Normal approximation using predicted standard deviation:

Exploration2.1B.14

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

[1] 0.01326

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


If X ˜ N(0.5,0.0132639527269323), then

P(X <= 0.55) = P(Z <= 3.77) = 0.9999
P(X >  0.55) = P(Z >  3.77) = 1e-04
[1] 8.175e-05
```



Approximate test for proportions with continuity correction:

Exploration2.1B.14b

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


1-sample proportions test with continuity correction

data:  x and n
X-squared = 14.19, df = 1, p-value = 8.262e-05
alternative hypothesis: true p is greater than 0.5
95 percent confidence interval:
 0.5282 1.0000
sample estimates:
     p
0.5503
```

Exact test for proportions:

Exploration2.1B.14c

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


Exact binomial test

data:  x and n
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:
 0.5282 1.0000
sample estimates:
probability of success
                0.5503
```

2.2 Inference for a Single Quantitative Variable

Example 2.2: Estimating Elapsed Time

Figure2.5

```
head(TimeEstimate)

  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 13.71 6.5 48       0

dotPlot(~Estimate, data = TimeEstimate, width = 1, cex = 0.25)
```

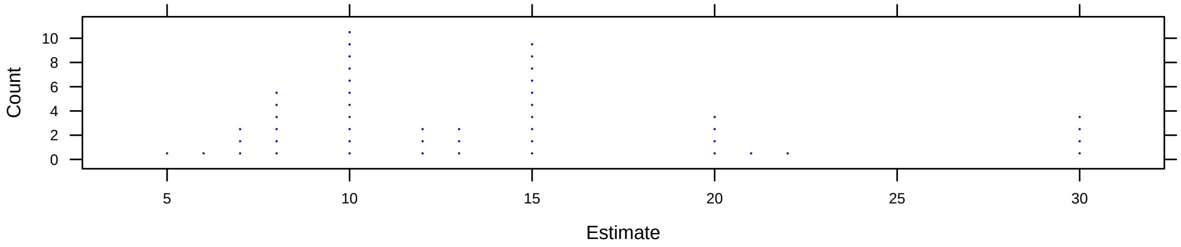


Figure2.6

```
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 6.49 6215       0

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

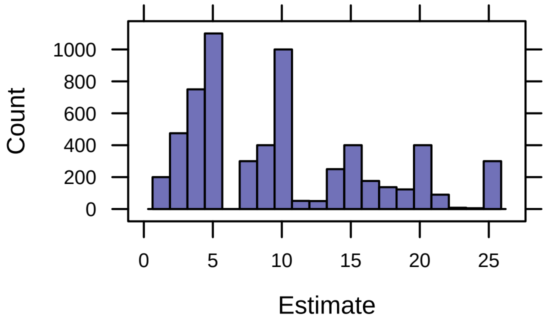


Figure2.7

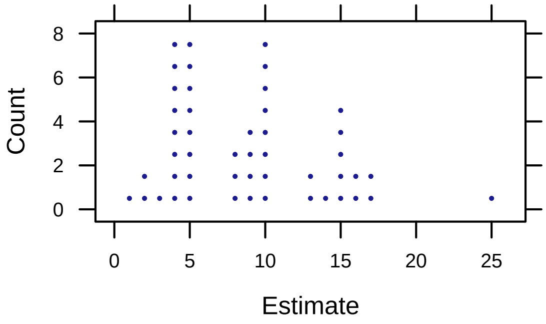
```
sample1 <- sample(TimePopulation, 48)
head(sample1, 3)
```

```
     Estimate orig.ids
1708        4     1708
2188       10     2188
5403       25     5403
```

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

```
 min   Q1 median Q3 max  mean    sd  n missing
   1 4.75      9 13  25 8.875 5.168 48       0
```

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

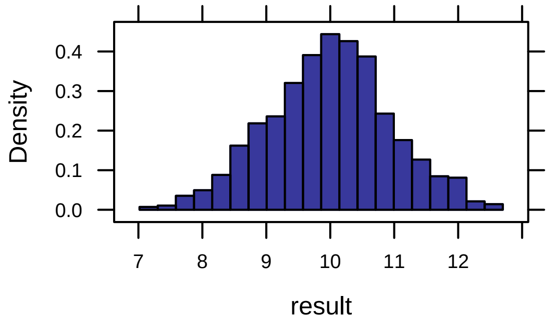


1. H_0: $\mu = 10$
   H_a: $\mu \neq 10$
   Test statistic: $\bar{x} = 13.71$ (the sample mean)
2. We simulate random samples from a finite population:

Figure2.8

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

```
  result
1  8.896
2  9.875
3 10.729

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



3. Strength of evidence:

Figure2.8b

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

TRUE
   0
```

Strength of evidence with the standardized statistic:

Figure2.8c

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

[1] 9.994

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

[1] 0.9617

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


If X ~ N(10,0.961742964079301), then

P(X <= 13.71) = P(Z <= 3.858) = 0.9999
P(X >  13.71) = P(Z >  3.858) = 1e-04
[1] 5.726e-05
```

Theory-based approach: One-sample t-test

Example2.2

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

[1] 3.954
```

Figure2.9

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

Warning:  NaNs produced
Warning:  NaNs produced
Warning:  NaNs produced
Warning:  NaNs produced
Warning:  NaNs produced
Warning:  NaNs produced
Warning:  NaNs produced
Warning:  NaNs produced
Warning:  NaNs produced
Warning:  NaNs produced
Warning:  NaNs produced
Warning:  NaNs produced
Warning:  NaNs produced
```

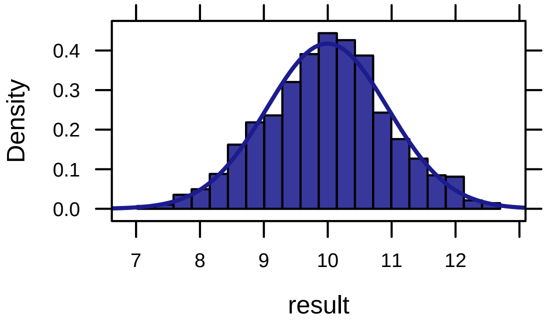


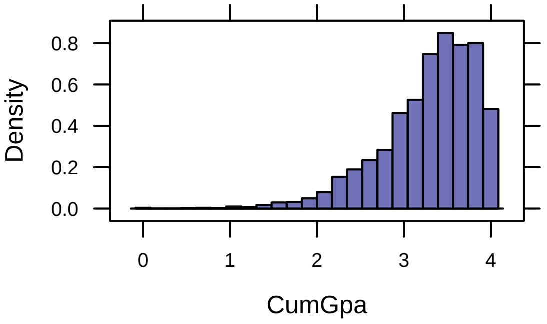
Figure2.10

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

[1] 0.0002571
```

Alternative Analysis: What about the median?

Figure2.11

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


  result
1      9
2      8
3     10


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


 TRUE
0.111
```



Exploration 2.2: Sleepless Nights?

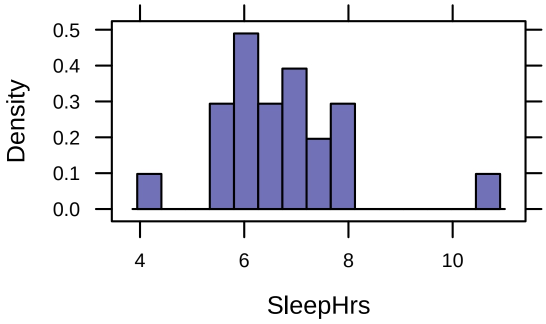
Exploration2.2.1

```
head(SleepTimes, 3)


  SleepHrs
1      7.0
2      5.5
3      8.0
```

Shape

Exploration2.2.10

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



Center

Exploration2.2.11

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

[1] 6.705
```

Exploration2.2.16

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

[1] 6.5
```

Variability

Exploration2.2.18

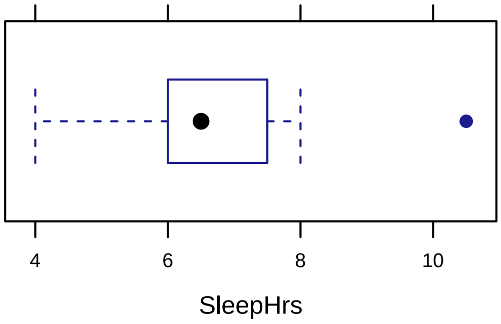
```
sd(~SleepHrs, data = SleepTimes)

[1] 1.297
```

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.

Exploration2.2.20

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



2.3 Errors and Significance

Example 2.3: Heart Transplant Operations (continued)

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 $\pi = 0.25$:

Exploration2.3.4

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

     n heads tails   prop
1 2124   539  1585 0.2538
2 2124   551  1573 0.2594
3 2124   535  1589 0.2519
```

3. Strength of evidence:

Exploration2.3.4b

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

TRUE
   0
```

Approximate test for proportions:

Exploration2.3.5

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


1-sample proportions test with continuity correction

data:  x and n
X-squared = 79.11, df = 1, p-value < 2.2e-16
```

```
alternative hypothesis: true p is greater than 0.25
95 percent confidence interval:
 0.317 1.000
sample estimates:
     p
0.3338
```

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

Exploration2.3.12

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

```
1-sample proportions test with continuity correction

data:  x and n
X-squared = 0.4267, df = 1, p-value = 0.2568
alternative hypothesis: true p is greater than 0.25
95 percent confidence interval:
 0.1974 1.0000
sample estimates:
  p
0.3
```

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

Exploration2.3.16

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

```
1-sample proportions test with continuity correction

data:  x and n
X-squared = 0.0905, df = 1, p-value = 0.6182
alternative hypothesis: true p is greater than 0.33
95 percent confidence interval:
 0.1974 1.0000
sample estimates:
  p
0.3
```

3

Estimation: How Large is the Effect?

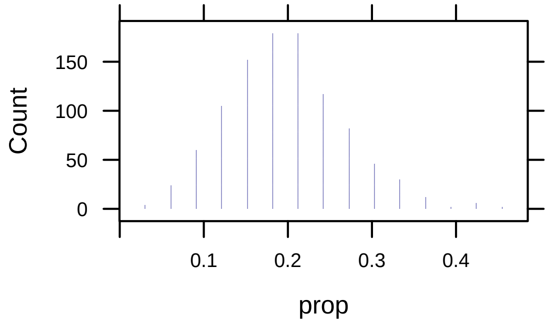
3.1 Statistical Inference - Confidence Intervals

Example 3.1: Can Dogs Sniff Out Cancer?

1. H_0: $\pi = 0.20$; H_a: $\pi > 0.20$
2. Test statistic: $\hat{p} = 0.909$ (the sample proportion of 30/33)
3. We simulate a world in which $\pi = 0.20$:

Figure3.1

```
simulation.cancer <- do(1000) * rflip(33, 0.2)

Loading required package:  parallel

head(simulation.cancer, 3)

   n heads tails   prop
1 33     8    25 0.2424
2 33     4    29 0.1212
3 33    10    23 0.3030

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

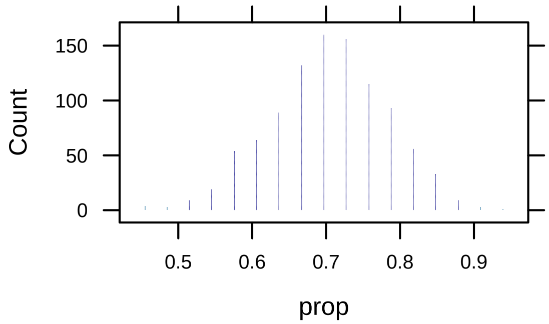


```
favstats(~prop, data = simulation.cancer)

    min     Q1 median     Q3    max   mean      sd    n missing
 0.0303 0.1515 0.1818 0.2424 0.4545 0.1966 0.06984 1000       0

prop(~(prop >= 0.909), data = simulation.cancer)

TRUE
   0
```

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

Figure3.2

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

   n heads tails   prop
1 33    23    10 0.6970
2 33    27     6 0.8182
3 33    25     8 0.7576

dotPlot(~prop, data = simulation.cancer2, groups = (prop <= 0.491 | prop >= 0.909), width = 0.001)
```



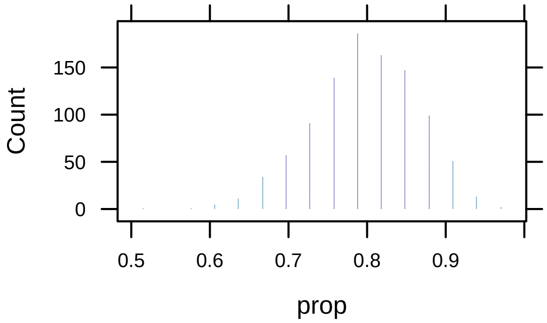
```
favstats(~prop, data = simulation.cancer2)

    min     Q1 median     Q3    max   mean      sd    n missing
 0.4545 0.6667  0.697 0.7576 0.9394 0.7035 0.07901 1000       0

prop(~(prop <= 0.491 | prop >= 0.909), data = simulation.cancer2)

 TRUE
0.011
```

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

3. We simulate a world in which $\pi = 0.80$:

Figure3.3

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

   n heads tails   prop
1 33    27     6 0.8182
2 33    25     8 0.7576
3 33    28     5 0.8485

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



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

    min     Q1 median     Q3    max   mean      sd    n missing
 0.5152 0.7576 0.7879 0.8485 0.9697 0.7972 0.06716 1000       0

prop(~(prop <= 0.691 | prop >= 0.909), data = simulation.cancer3)

 TRUE
0.118
```

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

Table3.1

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

p.value
 0.5007

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

p.value
 0.4474

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

```
p.value
 0.2272


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


p.value
 0.1442


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


p.value
0.07564


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


p.value
0.02793


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


p.value
0.00436
```

Exploration 3.1: Kissing Right?

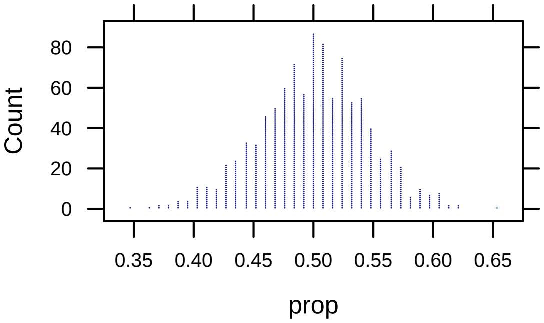
1. H_0: $\pi = 0.5$; H_a: $\pi > 0.5$

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

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

Exploration3.1.7

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

    n heads tails   prop
1 124    66    58 0.5323
2 124    65    59 0.5242
3 124    63    61 0.5081

dotPlot(˜prop, data = simulation.kiss, groups = (prop >= 0.645), width = 0.001)
```



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


    min     Q1 median     Q3    max   mean      sd    n missing
 0.3468 0.4677    0.5 0.5323 0.6532 0.4999 0.04471 1000       0

prop(~(prop >= 0.645), data = simulation.kiss)

 TRUE
0.001
```

4. Approximate test for proportions:

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



1-sample proportions test with continuity correction

data:  x and n
X-squared = 9.879, df = 1, p-value = 0.0008359
alternative hypothesis: true p is greater than 0.5
95 percent confidence interval:
 0.568 1.000
sample estimates:
     p
0.6452
```

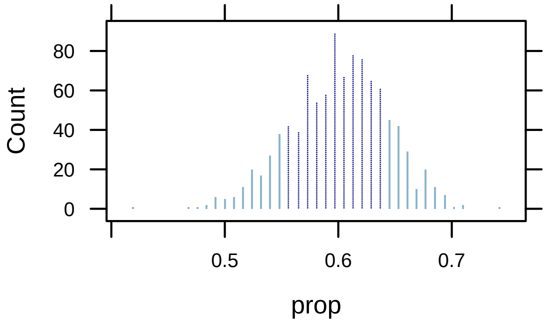
5. Exact test for proportions:

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



Exact binomial test

data:  x and n
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:
 0.5684 1.0000
sample estimates:
probability of success
                0.6452
```

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

Exploration3.1.8

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


    n heads tails   prop
1 124    78    46 0.6290
2 124    81    43 0.6532
3 124    82    42 0.6613

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



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

    min     Q1 median    Q3    max   mean      sd    n missing
 0.4194 0.5726 0.6048 0.629 0.7419 0.6008 0.04256 1000       0

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

 TRUE
0.303
```

4. Approximate test for proportions:

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



1-sample proportions test with continuity correction

data:  x and n
X-squared = 0.874, df = 1, p-value = 0.3499
alternative hypothesis: true p is not equal to 0.6
95 percent confidence interval:
 0.5536 0.7276
sample estimates:
     p
0.6452
```

5. Exact test for proportions:

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


Exact binomial test

data:  x and n
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:
 0.5542 0.7290
sample estimates:
probability of success
                0.6452
```

Exploration3.1.11

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


p.value
0.01915

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


p.value
0.03757

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


p.value
0.05778

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


p.value
 0.1024

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


p.value
 0.1465

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


p.value
 0.2355

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


p.value
 0.3151
```

Exploration3.1.11b

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

p.value
 0.2024

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

p.value
  0.114

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

p.value
0.07146

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

p.value
0.04242

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

p.value
 0.0185

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

 p.value
0.009269

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

 p.value
0.004281
```

Exploration3.1.13

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

probability of success                  lower                  upper
                0.6452                 0.5542                 0.7290
                 level
                0.9500
```

Exploration3.1.15

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

```
probability of success                    lower                  upper
                0.6452                   0.5265                 0.7524
                 level
                0.9900
```

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

Example 3.2: The Afforable 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% confidencen interval in its results.

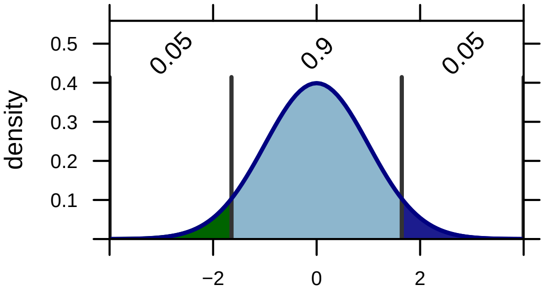
```
binom.test(713, 1034)  # 713 = 1034 * 0.69



Exact binomial test

data:  x and n
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:
 0.6604 0.7177
sample estimates:
probability of success
                0.6896
```

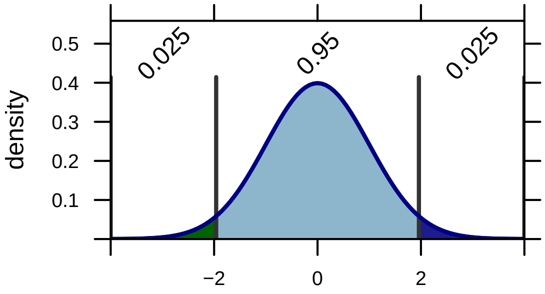
Theory-Based Approach

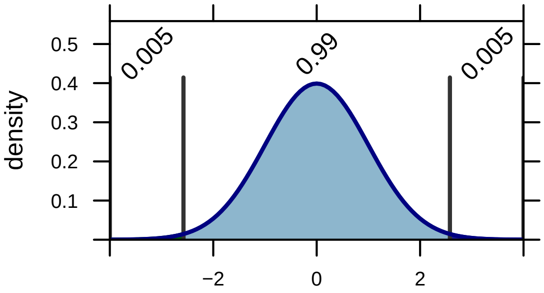
Figure3.6

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



If X ~ 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] 0.04998 0.95002


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



If X ~ N(0,1), then

P(X <= -1.96) = P(Z <= -1.96) = 0.025
  P(X <= 1.96) = P(Z <= 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] 0.025 0.975
```

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



If X ~ N(0,1), then

P(X <= -2.576) = P(Z <= -2.576) = 0.005
  P(X <= 2.576) = P(Z <= 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] 0.004998 0.995002
```







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

Figure3.7

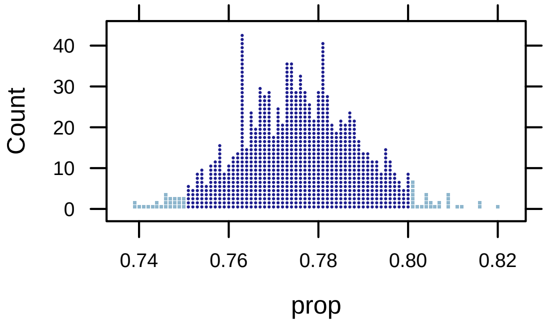
```
n <- 1034
p.hat <- 0.69
p.hat  # 0.69 = 713 / 1034


[1] 0.69


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


[1] 0.02819


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

```
[1] 0.6618

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

[1] 0.7182
```

Exploration 3.2: American Exceptionalism

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

```
simulation.amer <- do(1000) * rflip(1019, 0.775)

Loading required package:  parallel

head(simulation.amer, 3)

     n heads tails   prop
1 1019   797   222 0.7821
2 1019   787   232 0.7723
3 1019   783   236 0.7684

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



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

   min     Q1 median     Q3    max   mean      sd    n missing
 0.739 0.7674 0.7753 0.7851 0.8204 0.7757 0.01316 1000       0

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

 TRUE
0.052
```

4. Approximate test for proportions:

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


1-sample proportions test with continuity correction

data:  x and n
X-squared = 3.454, df = 1, p-value = 0.06308
alternative hypothesis: true p is not equal to 0.775
95 percent confidence interval:
 0.7736 0.8237
sample estimates:
     p
0.7998
```

5. Exact test for proportions:

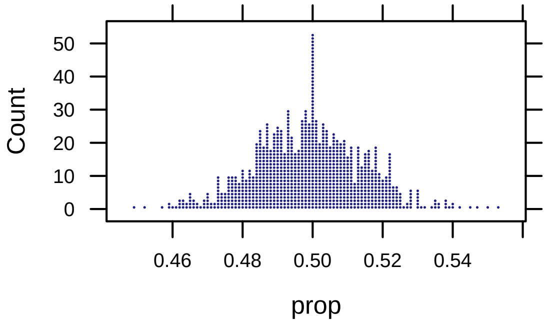
```
binom.test(815, 1019, p = 0.775)


Exact binomial test

data:  x and n
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:
 0.7739 0.8240
sample estimates:
probability of success
                0.7998
```

1. H_0: $\pi = 0.5$; H_a: $\pi \neq 0.5$

2. Test statistic: $\hat{p} = 0.80$ (the sample proportion of 815/1019)

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

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


     n heads tails   prop
1 1019   521   498 0.5113
2 1019   486   533 0.4769
3 1019   491   528 0.4818


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



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

    min     Q1 median     Q3    max  mean      sd    n missing
 0.4495 0.4887 0.4985 0.5093 0.5535 0.499 0.01546 1000       0

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

TRUE
   0
```

4. Approximate test for proportions:

```
prop.test(815, 1019)



1-sample proportions test with continuity correction

data:  x and n
X-squared = 365.2, df = 1, p-value < 2.2e-16
alternative hypothesis: true p is not equal to 0.5
95 percent confidence interval:
 0.7736 0.8237
sample estimates:
     p
0.7998
```

5. Exact test for proportions:

```
binom.test(815, 1019)



Exact binomial test

data:  x and n
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:
 0.7739 0.8240
sample estimates:
probability of success
                0.7998
```

Finding the standard deviation using simulated deviation:

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

[1] 0.01316

z <- (0.8 - 0.775)/sd
z

[1] 1.899

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


If X ~ N(0.775,0.0131649627564318), then

P(X <= 0.8) = P(Z <= 1.899) = 0.9712
P(X >  0.8) = P(Z >  1.899) = 0.0288
[1] 0.02878
```

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] 0.01316

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

[1] 0.02633

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

[1] 0.7737

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

[1] 0.8263
```

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] 0.01253
```

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


[1] 0.02506


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


[1] 0.7749


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


[1] 0.8251
```

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

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


[1] 0.01253


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


[1] 0.02456


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


[1] 0.7754


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


[1] 0.8246
```

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.

```
cdata(0.95, prop, data = simulation.amer)


      low        hi central.p
   0.7507    0.8008    0.9500
```

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

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



Exact binomial test
```

```
data:  x and n
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:
 0.7739 0.8240
sample estimates:
probability of success
                0.7998
```

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

```


Exact binomial test

data:  x and n
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:
 0.7656 0.8311
sample estimates:
probability of success
                0.7998
```

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

```


Exact binomial test

data:  x and n
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:
 0.7781 0.8203
sample estimates:
probability of success
                0.7998
```

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

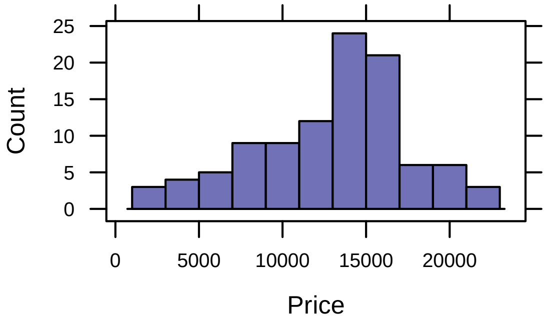
Example 3.3: Used Cars

```
head(UsedCars)
```

Figure3.9

```
  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  13992 15999 21990 13292 4535 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

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

[1] 4535

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

[1] 898

mean - MoE             # lower limit of 95% CI

[1] 12394

mean + MoE             # upper limit of 95% CI

[1] 14190
```

Theory-based approach

Figure3.10

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


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

Figure3.11

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


mean of x     lower     upper     level
  13292.3   12547.0   14037.7       0.9


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)

Exploration3.3.1

```
head(SleepTimes)


  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 6.705 1.297 22       0
```

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

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


[1] 22


mean <- mean(~ SleepHrs, data = SleepTimes); mean


[1] 6.705


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

```
[1] 1.297

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

[1] 0.5531

mean - MoE            # lower limit of 95% CI

[1] 6.151

mean + MoE            # upper limit of 95% CI

[1] 7.258
```

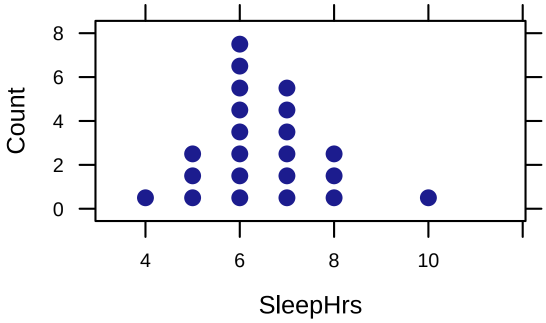
Theory-based approach

Exploration3.3.8

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

mean of x     lower     upper     level
    6.705     6.129     7.280     0.950
```

Exploration3.3.9

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



3.4 Factors That Affect the Width of a Confidence Interval

Example 3.4: The Afforable Care Act (continued)

Table3.5

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

probability of success                  lower                  upper
                0.6896                 0.6650                 0.7133
                 level
                0.9000

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

probability of success                  lower                  upper
                0.6896                 0.6604                 0.7177
                 level
                0.9500

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

probability of success                  lower                  upper
                0.6896                 0.6512                 0.7262
                 level
                0.9900
```

Sample size

Figure3.12

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

probability of success                  lower                  upper
                0.7000                 0.6002                 0.7876
                 level
                0.9500

confint(binom.test(140, 200))

probability of success                  lower                  upper
                0.7000                 0.6314                 0.7626
                 level
                0.9500

confint(binom.test(280, 400))

probability of success                  lower                  upper
                0.7000                 0.6525                 0.7445
                 level
                0.9500
```

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()`.

Figure3.13

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


probability of success                  lower                   upper
                0.8104                 0.7852                  0.8339
                 level
                0.9500


MoE838 <- 0.8339078 - 0.7852004
MoE838


[1] 0.04871

confint(binom.test(196, 1034))


probability of success                  lower                   upper
                0.1896                 0.1661                  0.2148
                 level
                0.9500


MoE196 <- 0.2147996 - 0.1660922
MoE196


[1] 0.04871
```

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] 9.307


mean - MoE              # lower limit of 95% CI


[1] 694.7


mean + MoE              # upper limit of 95% CI


[1] 713.3
```

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


[1] 18.61


mean - MoE            # lower limit of 95% CI


[1] 685.4


mean + MoE            # upper limit of 95% CI


[1] 722.6
```

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] 27.47


mean - MoE            # lower limit of 95% CI


[1] 676.5


mean + MoE            # upper limit of 95% CI


[1] 731.5
```

Exploration 3.4B: Reese's Pieces

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

Exploration3.4B.4

```
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 0.408 0.6114     0.51   Yes      1
```

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

Exploration3.4B.5

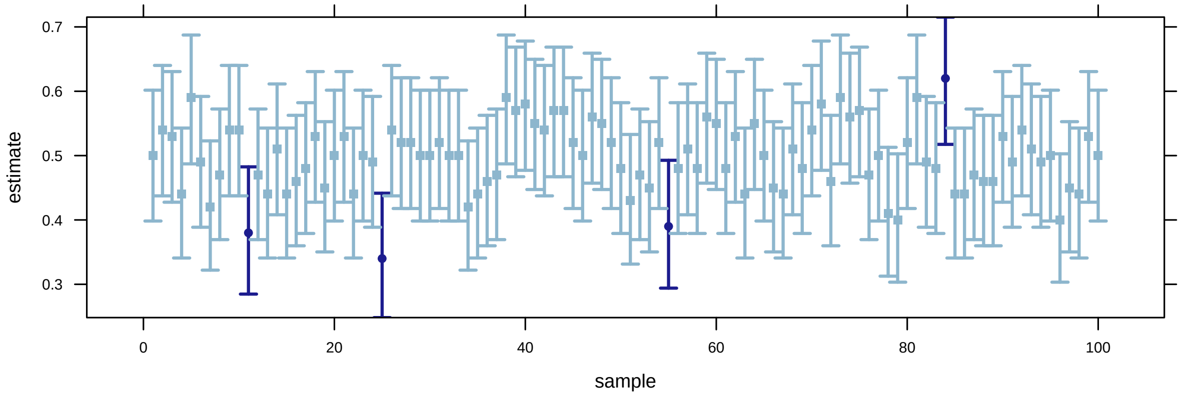
```
simulation.CI <- CIsim(100, samples = 100, rdist = rbinom, args = list(size = 1, prob = 0.5),
    method = binom.test, method.args = list(success = 1), verbose = FALSE, estimand = 0.5)
```

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

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


  No
0.04
```

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

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



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

```
simulation.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)
```

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

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


   No
0.024
```

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

```
simulation.CI3 <- CIsim(100, samples = 1000, rdist = rbinom, args = list(size = 1, prob = 0.5),
    conf.level = 0.9, method = binom.test, method.args = list(success = 1), verbose = FALSE,
    estimand = 0.5)
```

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

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


   No
0.097
```

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

```
simulation.CI4 <- CIsim(400, samples = 100, rdist = rbinom, args = list(size = 1, prob = 0.5),
    conf.level = 0.9, method = binom.test, method.args = list(success = 1), verbose = FALSE,
    estimand = 0.5)
```

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

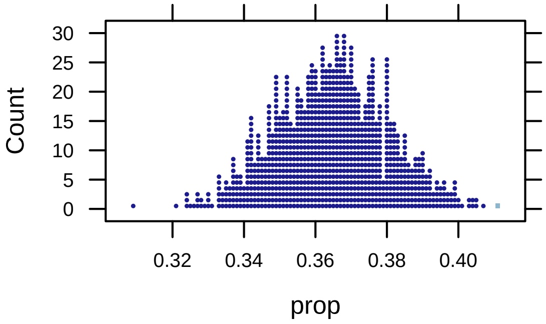
```
prop(~cover, data = simulation.CI4)


  No
0.11
```

3.5 Cautions When Conducting Inference

1. H_0: $\pi = 0.3645$; H_a: $\pi > 0.3645$
2. Test statistic: $\hat{p} = 0.41$ (the sample proportion)
3. We simulate a world in which $\pi = 0.3645$:

Figure3.14

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


     n heads tails  prop
1 1000   376   624 0.376
2 1000   403   597 0.403
3 1000   326   674 0.326

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



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

   min    Q1 median     Q3   max   mean      sd    n missing
 0.309 0.353  0.365 0.3752 0.411 0.3646 0.01586 1000       0

prop(~(prop >= 0.41), data = simulation.obama)

 TRUE
0.001
```

Exploration 3.5A: Voting for President

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

Exploration3.5A.3

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

probability of success                     lower                    upper
                0.6824                    0.6584                   0.7057
                 level
                0.9900
```

Another famous case of problems in Presidential election polling

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

Exploration3.5A.9

```
confint(binom.test(1368000, 2400000, conf.level = 0.999))  # 1368000 = 2400000 * 0.57

probability of success                     lower                    upper
                0.5700                    0.5689                   0.5711
                 level
                0.9990
```

Example 3.5B: Parapsychology Studies (continued)

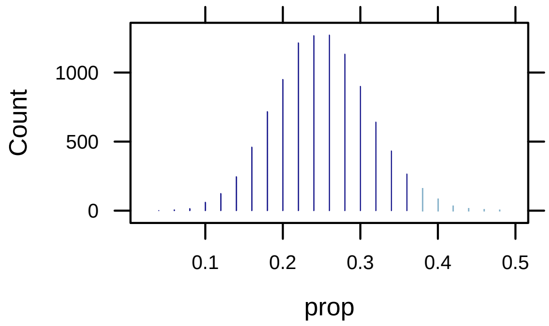
Example3.5B

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

probability of success                     lower                    upper
                0.3338                    0.3138                   0.3543
                 level
                0.9500

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

```
probability of success                        lower                    upper
                0.3338                       0.3076                   0.3607
                 level
                0.9900
```

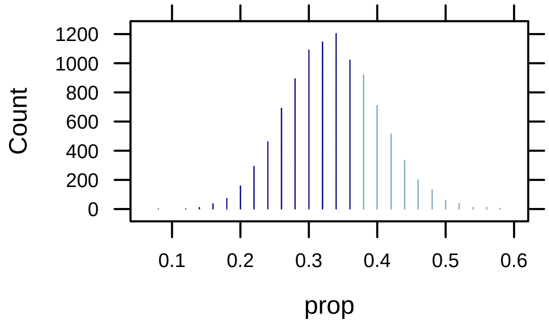
1. H_0: $\pi = 0.25$; H_a: $\pi > 0.25$
2. Test statistic: $\hat{p} = 0.38$ (the sample proportion of 19/50)
3. We simulate a world in which $\pi = 0.25$:

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

   n heads tails prop
1 50    18    32 0.36
2 50    11    39 0.22
3 50     9    41 0.18

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

  TRUE
0.0296
```



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

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

   n heads tails prop
1 50    19    31 0.38
2 50    14    36 0.28
3 50    17    33 0.34

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

  TRUE
0.2912
```

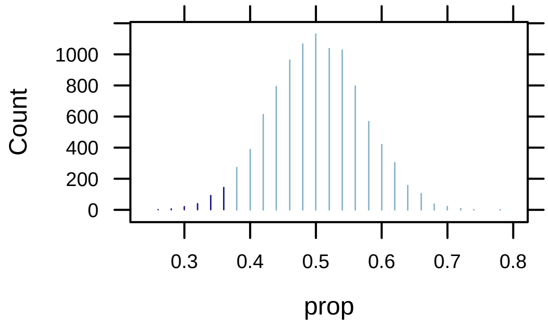


1. H_0: $\pi = 1/2$; H_a: $\pi > 1/2$

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

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

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

   n heads tails prop
1 50    27    23 0.54
2 50    28    22 0.56
3 50    23    27 0.46

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

  TRUE
0.9689
```



3.5.1 Exploration 3.5B: Cat Households

1. H_0: $\pi = 1/3$; H_a: $\pi < 1/3$

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

3. Exact test for proportions:

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


Exact binomial test

data:  x and n
number of successes = 15228, number of trials = 47000, p-value = 8.654e-06
alternative hypothesis: true probability of success is less than 0.3333
99.9 percent confidence interval:
 0.0000 0.3307
sample estimates:
probability of success
                 0.324

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


Exact binomial test

data:  x and n
number of successes = 15228, number of trials = 47000, p-value = 8.654e-06
alternative hypothesis: true probability of success is less than 0.3333
95 percent confidence interval:
 0.0000 0.3276
sample estimates:
probability of success
                 0.324
```

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

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

    n heads tails prop
1 100    36    64 0.36
2 100    39    61 0.39
3 100    37    63 0.37
```

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 = simulation.pets)

      low        hi central.p
     0.23      0.43      0.95
```

1. H_0: $\pi = 0.30$; H_a: $\pi < 0.30$

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

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

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

```
    n heads tails prop
1 100    23    77 0.23
2 100    34    66 0.34
3 100    30    70 0.30

prop(~(prop <= 0.243), data = simulation.pets2)

 TRUE
0.128
```

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

     low        hi central.p
    0.22      0.38      0.90
```

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


probability of success                  lower                   upper
                0.3300                 0.2392                  0.4312
                 level
                0.9500


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



Exact binomial test

data:  x and n
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:
 0.0000 0.3206
sample estimates:
probability of success
                  0.24


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


probability of success                  lower                   upper
                0.3300                 0.2523                  0.4155
                 level
                0.9000


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



Exact binomial test

data:  x and n
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:
 0.000 0.314
sample estimates:
probability of success
                  0.25


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


probability of success                  lower                  upper
                0.3340                 0.2927                 0.3772
                 level
                0.9500


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



Exact binomial test

data:  x and n
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:
 0.0000 0.3273
sample estimates:
probability of success
                 0.292


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


probability of success                  lower                  upper
                0.3300                 0.2392                 0.4312
                 level
                0.9500


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



Exact binomial test

data:  x and n
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:
 0.0000 0.3206
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
```

```
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
```

Table4.1

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


         light
nearsight Darkness NightLight RoomLight
     Near   0.1047     0.3362    0.5467
     Not    0.8953     0.6638    0.4533


tally(~nearsight | light, data = NightLight)


         light
nearsight Darkness NightLight RoomLight
     Near   0.1047     0.3362    0.5467
     Not    0.8953     0.6638    0.4533


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
```

Exploration 4.1: Home Court Disadvantage?

4.2 Observational studies versus experiments

Example 4.2: Lying on the Internet

Exploration 4.2: Have a Nice Trip

```
sim <- do(2) * rflip(12, 16/24)
sim


   n heads tails prop
1 12     6     6  0.5
2 12     6     6  0.5
```

5

Comparing Two Proportions

5.1 Comparing Two Groups: Categorical Response

Example 5.1: Good and Bad Perceptions

Table5.1

```
head(GoodandBad, 30)
```

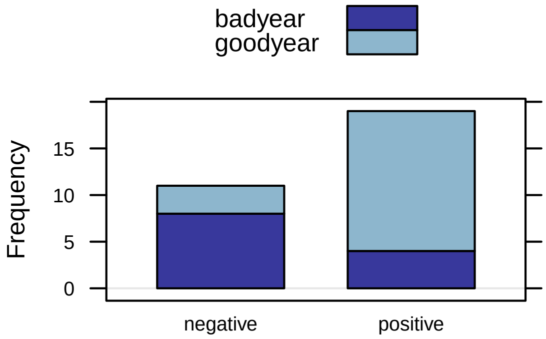
```
    Wording Perception
1  goodyear   positive
2  goodyear   negative
3   badyear   positive
4  goodyear   positive
5  goodyear   negative
6   badyear   positive
7  goodyear   positive
8  goodyear   positive
9  goodyear   positive
10  badyear   negative
11 goodyear   negative
12  badyear   negative
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
```

Table5.2

```
tally(~Perception + Wording, data = GoodandBad, margins = TRUE)


          Wording
Perception badyear goodyear Total
  negative       8        3    11
  positive       4       15    19
  Total         12       18    30


tally(Perception ~ Wording, data = GoodandBad)


          Wording
Perception badyear goodyear
  negative  0.6667   0.1667
  positive  0.3333   0.8333


prop(Perception ~ Wording, data = GoodandBad)


 negative.badyear negative.goodyear
           0.6667            0.1667


prop(Perception ~ Wording, level = "positive", data = GoodandBad)


 positive.badyear positive.goodyear
           0.3333            0.8333
```

Figure5.1

```
bargraph(~Perception, groups = Wording, data = GoodandBad, stack = TRUE, auto.key = TRUE)
```



Summarizing the data

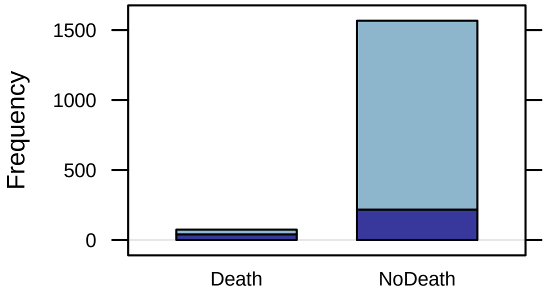
Exploration 5.1: Murderous Nurse?

```
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")
  )
```

```
tally(~patient + shift, data = Nurse, margins = TRUE)


         shift
patient   Gilbert NoGilbert Total
  Death        40        34    74
  NoDeath     217      1350  1567
  Total       257      1384  1641


tally(patient ~ shift, data = Nurse)  # conditional prop


         shift
patient   Gilbert NoGilbert
  Death   0.15564   0.02457
  NoDeath 0.84436   0.97543
```

Exploration 5.1.10

```
bargraph(~patient, groups = shift, data = Nurse, stack = TRUE)
```



Exploration 5.1.14

```
prop(patient ~ shift, data = Nurse)


  Death.Gilbert Death.NoGilbert
        0.15564         0.02457


diff(prop(patient ~ shift, data = Nurse))


Death.NoGilbert
        -0.1311
```

Further Analysis

```
Nurse2 <- rbind(
  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")
   )
```

Exploration 5.1.

```
tally(~patient + shift, data = Nurse2, margin = TRUE)
```

```
         shift
patient   Gilbert NoGilbert Total
  Death       100       357   457
  NoDeath     157      1027  1184
  Total       257      1384  1641
```

```
tally(patient ~ shift, data = Nurse2)
```

```
         shift
patient   Gilbert NoGilbert
  Death     0.3891    0.2579
  NoDeath   0.6109    0.7421
```

```
diff(prop(patient ~ shift, data = Nurse2))  # diff in conditional prop
```

```
Death.NoGilbert
        -0.1312
```

```
# relative risk
```

5.2 Comparing Two Properties: Simulation-Based Approach

Example 5.2: Swimming with Dolphins

Table5.3

```
head(Dolphin)
```

```
  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 Total
  Improve          3      10    13
  NotImprove      12       5    17
  Total           15      15    30
```

```
tally(Response ~ Swimming, data = Dolphin)
```

```
            Swimming
Response     Control Dolphin
  Improve     0.2000  0.6667
  NotImprove  0.8000  0.3333
```

```
diff(prop(Response ~ Swimming, data = Dolphin))
```

```
Improve.Dolphin
         0.4667
```

```
bargraph(~Swimming, data = Dolphin, groups = Response, stack = TRUE, auto.key = TRUE)
```

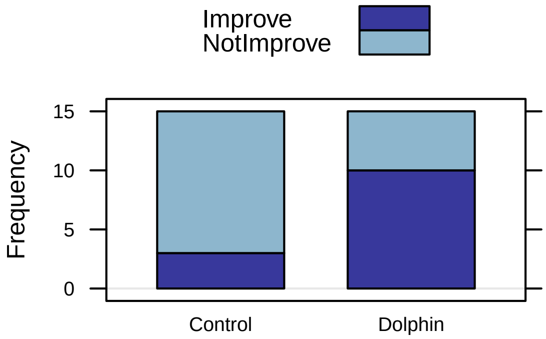


Figure5.4

```
tally(~shuffle(Response) + Swimming, data = Dolphin, margins = TRUE)
```

```
                 Swimming
shuffle(Response) Control Dolphin Total
       Improve          5       8    13
       NotImprove      10       7    17
       Total           15      15    30
```

```
tally(~shuffle(Response) + Swimming, data = Dolphin, margins = TRUE)
```

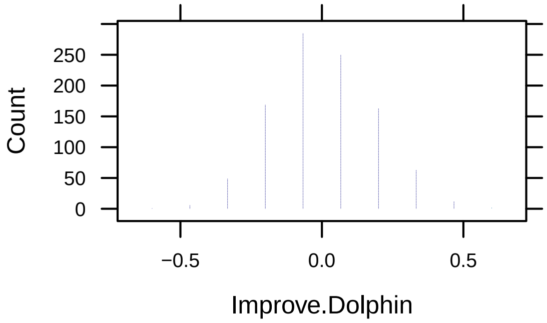
```
                 Swimming
shuffle(Response) Control Dolphin Total
       Improve          5       8    13
       NotImprove      10       7    17
       Total           15      15    30
```

```
tally(~shuffle(Response) + Swimming, data = Dolphin, margins = TRUE)
```

```
                 Swimming
shuffle(Response) Control Dolphin Total
       Improve          7       6    13
       NotImprove       8       9    17
       Total           15      15    30


diff(prop(Response ~ Swimming, data = Dolphin))


Improve.Dolphin
         0.4667


diff(prop(shuffle(Response) ~ Swimming, data = Dolphin))


Improve.Dolphin
       -0.06667
```

1. H_0: $\pi_{dolphins} - \pi_{control} = 0$; H_a: $\pi_{dolphins} - \pi_{control} > 0$
2. Test statistic: $\hat{p}_{dolphins} - \hat{p}_{control} = 0.4667$ (the difference in the conditional sample proportions)
3. We simulate a world in which $\pi_{dolphins} - \pi_{control} = 0$:

```
simulation.dol <- do(1000) * diff(prop(shuffle(Response) ~ Swimming, data = Dolphin))
head(simulation.dol, 3)

  Improve.Dolphin
1        -0.20000
2         0.06667
3         0.20000

dotPlot(~Improve.Dolphin, data = simulation.dol, groups = (Improve.Dolphin >= 0.4667), width = 1/15)
```



```
favstats(~Improve.Dolphin, data = simulation.dol)

  min       Q1   median      Q3 max     mean    sd    n missing
 -0.6 -0.06667 -0.06667 0.06667 0.6 0.004533 0.182 1000       0

prop(~(Improve.Dolphin >= 0.4667), data = simulation.dol)

 TRUE
0.002
```

4. Normal approximation:

5. Approximate test for difference in proportions:

```
prop.test(Response ~ Swimming, data = Dolphin)


2-sample test for equality of proportions with continuity correction

data:  t(table_from_formula)
X-squared = 4.887, df = 1, p-value = 0.02706
alternative hypothesis: two.sided
95 percent confidence interval:
 -0.84620 -0.08713
sample estimates:
prop 1 prop 2
0.2000 0.6667
```

Estimation

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

```
# given difference in sample proportions
diff <- diff(prop(Response ~ Swimming, data = Dolphin))
# simulated standard deviation
sd <- sd(~Improve.Dolphin, data = simulation.dol)
# margin of error for 95% CI
MoE <- 2 * sd
MoE


[1] 0.3639


# lower limit of 95% CI
diff - MoE


Improve.Dolphin
         0.1027


# upper limit of 95% CI
diff + MoE


Improve.Dolphin
         0.8306
```

Determining a 95% confidence interval using the approximate test for proportions:

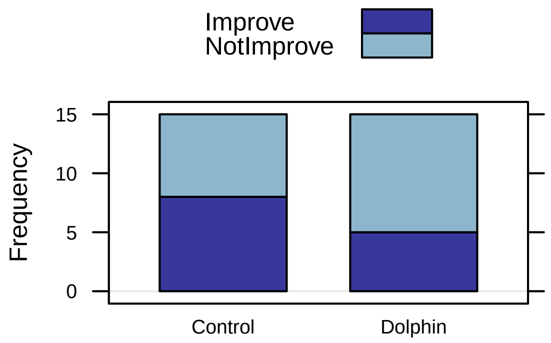
```
confint(prop.test(Response ~ Swimming, data = Dolphin))


  prop 1   prop 2    lower    upper    level
 0.20000  0.66667 -0.84620 -0.08713  0.95000
```

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")
   )
```

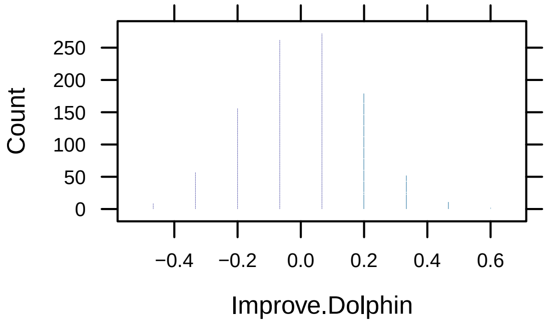
```
tally(~Response + Swimming, data = Dolphin2, margin = TRUE)


            Swimming
Response     Control Dolphin Total
  Improve          8       5    13
  NotImprove       7      10    17
  Total           15      15    30


diff(prop(Response ~ Swimming, data = Dolphin2))


Improve.Dolphin
           -0.2
```

```
bargraph(~Swimming, data = Dolphin2, groups = Response, stack = TRUE, auto.key = TRUE)
```



1. H_0: $\pi_{dolphins} - \pi_{control} = 0$; H_a: $\pi_{dolphins} - \pi_{control} > 0$
2. Test statistic: $\hat{p}_{dolphins} - \hat{p}_{control} = 0.20$ (the difference in the conditional sample proportions)
3. We simulate a world in which $\pi_{dolphins} - \pi_{control} = 0$:

```
simulation.dol2 <- do(1000) * diff(prop(shuffle(Response) ~ Swimming, data = Dolphin2))
head(simulation.dol2, 3)


  Improve.Dolphin
1         0.06667
2        -0.20000
3         0.20000


dotPlot(~Improve.Dolphin, data = simulation.dol2, groups = (Improve.Dolphin >=0.20),
        width = 1/15)
```



```
favstats(~Improve.Dolphin, data = simulation.dol2)

     min       Q1  median      Q3 max     mean     sd    n missing
 -0.4667 -0.06667 0.06667 0.06667 0.6 0.005733 0.1815 1000       0

prop(~(Improve.Dolphin >= 0.2), data = simulation.dol2)

 TRUE
0.244
```

4. Approximate test for difference in proportions:

```
prop.test(Response ~ Swimming, data = Dolphin2, alt = "greater")


2-sample test for equality of proportions with continuity correction

data:  t(table_from_formula)
X-squared = 0.543, df = 1, p-value = 0.2306
alternative hypothesis: greater
95 percent confidence interval:
 -0.1582  1.0000
sample estimates:
prop 1 prop 2
0.5333 0.3333
```

or, without having to create a dataframe:

```
success <- c(8, 5)
n <- c(15, 15)
prop.test(success, n, alt = "greater")


2-sample test for equality of proportions with continuity correction

data:  x and n
X-squared = 0.543, df = 1, p-value = 0.2306
alternative hypothesis: greater
95 percent confidence interval:
 -0.1582  1.0000
sample estimates:
prop 1 prop 2
0.5333 0.3333
```

Relative Risk

Exploration 5.2: Is Yawning Contagious?

Exploration5.2.9

```
head(Yawning, 3)
```

```
  YawnSeed Response
1   Seeded     Yawn
2   Seeded     Yawn
3   Seeded     Yawn
```

```
tally(~Response + YawnSeed, data = Yawning, margin = TRUE)
```

```
        YawnSeed
Response Control Seeded Total
  NoYawn      13     23    36
  Yawn         3     11    14
  Total       16     34    50
```

Exploration5.2.10

```
tally(Response ~ YawnSeed, data = Yawning)
```

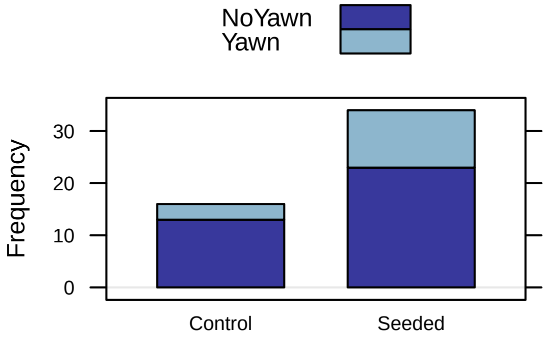
```
        YawnSeed
Response Control Seeded
  NoYawn  0.8125 0.6765
  Yawn    0.1875 0.3235
```

```
diff(prop(Response ~ YawnSeed, level = "Yawn", data = Yawning))
```

```
Yawn.Seeded
      0.136
```

Exploration5.2.11

```
bargraph(~YawnSeed, data = Yawning, groups = Response, stack = TRUE, auto.key = TRUE)
```



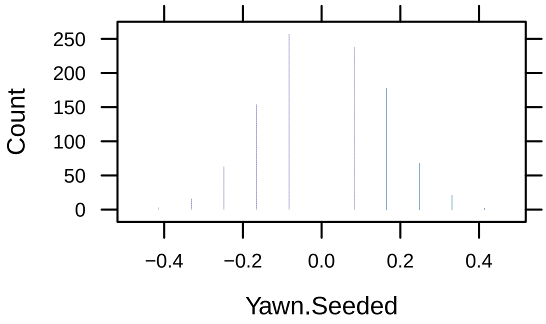
Exloration5.2.14

```
tally(~shuffle(Response) + YawnSeed, data = Yawning, margins = TRUE)


                 YawnSeed
shuffle(Response) Control Seeded Total
          NoYawn       10     26    36
          Yawn          6      8    14
          Total        16     34    50
```

1. H_0: $\pi_{seeded} - \pi_{control} = 0$; H_a: $\pi_{seeded} - \pi_{control} > 0$
2. Test statistic: $\hat{p}_{seeded} - \hat{p}_{control} = 0.136$ (the difference in the conditional sample proportions)
3. We simulate a world in which $\pi_{seeded} - \pi_{control} = 0$:

Exploration5.2.16

```
simulation.yawn <-
  do(1000) * diff(prop(shuffle(Response) ~ YawnSeed, level = "Yawn", data = Yawning))
head(simulation.yawn, 3)

  Yawn.Seeded
1    -0.23162
2     0.04412
3     0.04412

dotPlot(~Yawn.Seeded, data = simulation.yawn, groups = (Yawn.Seeded >= 0.136))
```



```
favstats(~Yawn.Seeded, data = simulation.yawn)

     min       Q1  median    Q3    max     mean     sd    n missing
 -0.4154 -0.04779 0.04412 0.136 0.4118 0.002941 0.1377 1000       0

prop(~(Yawn.Seeded >= 0.136), data = simulation.yawn)

 TRUE
0.269
```

4. Approximate test for difference in proportions:

```
prop.test(Response ~ YawnSeed, data = Yawning, alt = "greater")

Warning:  Chi-squared approximation may be incorrect


2-sample test for equality of proportions with continuity correction

data:  t(table_from_formula)
X-squared = 0.4379, df = 1, p-value = 0.2541
alternative hypothesis: greater
95 percent confidence interval:
 -0.1177  1.0000
sample estimates:
prop 1 prop 2
0.8125 0.6765
```

```
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")
   )
```

```
head(Yawning2, 3)


  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 Total
  NoYawn      12     24    36
  Yawn         4     10    14
  Total       16     34    50
```

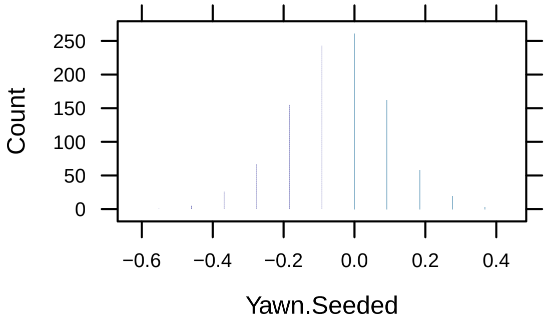
Exploration5.2.21

```
tally(Response ~ YawnSeed, data = Yawning2)


        YawnSeed
Response Control Seeded
  NoYawn  0.7500 0.7059
  Yawn    0.2500 0.2941


diff(prop(Response ~ YawnSeed, level = "Yawn", data = Yawning2))


Yawn.Seeded
    0.04412
```

1. H_0: $\pi_{seeded} - \pi_{control} = 0$; H_a: $\pi_{seeded} - \pi_{control} > 0$
2. Test statistic: $\hat{p}_{seeded} - \hat{p}_{control} = 0.0441$ (the difference in the conditional sample proportions)
3. We simulate a world in which $\pi_{seeded} - \pi_{control} = 0$:

Exploration5.2.23

```
simulation.yawn2 <-
  do(1000) * diff(prop(shuffle(Response) ~ YawnSeed, level = "Yawn", data = Yawning2))
head(simulation.yawn2, 3)

  Yawn.Seeded
1    -0.32353
2    -0.04779
3    -0.04779

dotPlot(~Yawn.Seeded, data = simulation.yawn2, groups = (Yawn.Seeded >= 0.0441))
```



```
favstats(~Yawn.Seeded, data = simulation.yawn2)

     min      Q1  median      Q3    max      mean     sd    n missing
 -0.5074 -0.1397 0.04412 0.04412 0.4118 -0.005699 0.1413 1000       0

prop(~(Yawn.Seeded >= 0.0441), data = simulation.yawn2)

 TRUE
0.503
```

4. Approximate test for difference in proportions:

```
prop.test(Response ~ YawnSeed, data = Yawning2, alt = "greater")


2-sample test for equality of proportions with continuity correction

data:  t(table_from_formula)
X-squared = 0, df = 1, p-value = 0.5
alternative hypothesis: greater
95 percent confidence interval:
 -0.2196  1.0000
sample estimates:
prop 1 prop 2
0.7500 0.7059
```

or, without having to create a dataframe:

```
success <- c(4, 10)
n <- c(16, 34)
prop.test(success, n, alt = "greater")


2-sample test for equality of proportions with continuity correction

data:  x and n
X-squared = 0, df = 1, p-value = 0.5
alternative hypothesis: greater
95 percent confidence interval:
 -0.3078  1.0000
sample estimates:
prop 1 prop 2
0.2500 0.2941
```

Estimation

Exploration5.2.24c

```
sd <- sd(~Yawn.Seeded, data = simulation.yawn2)
sd


[1] 0.1413
```

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

Exploration5.2.24d

```
# given difference in sample proportions
diff <- diff(prop(Response ~ YawnSeed, level = "Yawn", data = Yawning2))
# previously found simulated standard deviation
sd


[1] 0.1413


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


[1] 0.2827


# lower limit of 95% CI
diff - MoE


Yawn.Seeded
    -0.2386


# upper limit of 95% CI
diff + MoE


Yawn.Seeded
     0.3268
```

Determining a 95% confidence interval using the approximate test for proportions:

```
confint(prop.test(Response ~ YawnSeed, data = Yawning2))


 prop 1  prop 2   lower   upper   level
 0.7500  0.7059 -0.2617  0.3499  0.9500
```

Effect of Sample Size

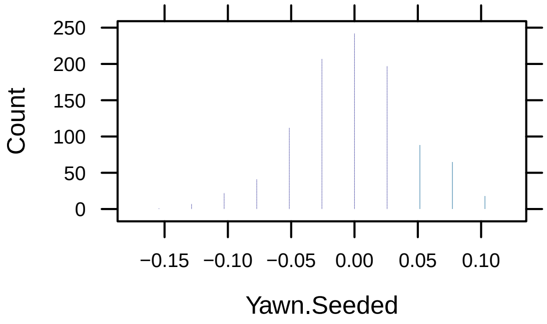
```
Yawning3 <- rbind(
  do(240)  *  data.frame(Response = "NoYawn", YawnSeed = "Control"),
  do(120)  *  data.frame(Response = "NoYawn", YawnSeed = "Seeded"),
  do(100)  *  data.frame(Response = "Yawn",   YawnSeed = "Control"),
  do(40)   *  data.frame(Response = "Yawn",   YawnSeed = "Seeded")
   )
```

```
head(Yawning3, 3)


  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 Total
  NoYawn     240    120   360
  Yawn       100     40   140
  Total      340    160   500
```

1. H_0: $\pi_{seeded} - \pi_{control} = 0$; H_a: $\pi_{seeded} - \pi_{control} > 0$
2. Test statistic: $\hat{p}_{seeded} - \hat{p}_{control} = 0.0441$ (the difference in the conditional sample proportions)
3. We simulate a world in which $\pi_{seeded} - \pi_{control} = 0$:

Exploration5.2.32

```
simulation.yawn3 <-
  do(1000) * diff(prop(shuffle(Response) ~ YawnSeed, level = "Yawn", data = Yawning3))
head(simulation.yawn3, 3)

  Yawn.Seeded
1    -0.14522
2    -0.02574
3    -0.01654

dotPlot(~Yawn.Seeded, data = simulation.yawn3, groups = (Yawn.Seeded >= 0.0441))
```



```
favstats(~Yawn.Seeded, data = simulation.yawn3)

     min       Q1    median      Q3    max        mean       sd    n missing
 -0.1452 -0.02574 0.001838 0.02941 0.1121 -0.0003033 0.04435 1000       0

prop(~(Yawn.Seeded >= 0.0441), data = simulation.yawn3)

 TRUE
0.171
```

4. Approximate test for difference in proportions:

```
prop.test(Response ~ YawnSeed, data = Yawning3, alt = "greater")


2-sample test for equality of proportions with continuity correction

data:  t(table_from_formula)
X-squared = 0.843, df = 1, p-value = 0.8207
alternative hypothesis: greater
95 percent confidence interval:
 -0.1182  1.0000
sample estimates:
prop 1 prop 2
0.7059 0.7500
```

or, without having to create a dataframe:

```
success <- c(40, 100)
n <- c(160, 340)
prop.test(success, n, alt = "greater")


2-sample test for equality of proportions with continuity correction

data:  x and n
X-squared = 0.843, df = 1, p-value = 0.8207
alternative hypothesis: greater
95 percent confidence interval:
 -0.1182  1.0000
sample estimates:
prop 1 prop 2
0.2500 0.2941
```

Relative risk

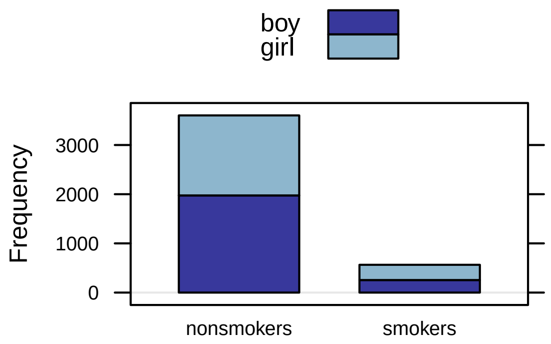
5.3 Comparing Two Proportions: Theory-Based Approach

Example 5.3: Smoking and Birth Gender

```
head(Smoking, 3)


  Parents Child
1 smokers  girl
2 smokers  girl
3 smokers  girl


summary(Smoking)


       Parents      Child
 nonsmokers:3602   boy  :2230
 smokers   : 565   girl:1937


tally(~Parents + Child, data = Smoking, margin = TRUE)


            Child
Parents        boy girl Total
  nonsmokers 1975 1627  3602
  smokers     255  310   565
  Total      2230 1937  4167
```

Figure5.9

```
bargraph(~Parents, data = Smoking, groups = Child, stack = TRUE, auto.key = TRUE)
```



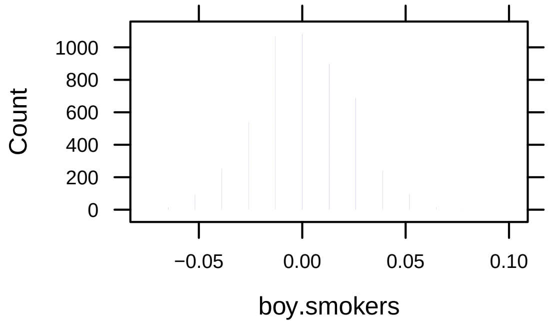
```
tally(Child ~ Parents, data = Smoking)


      Parents
Child  nonsmokers smokers
  boy      0.5483  0.4513
  girl     0.4517  0.5487
```

```
diff(prop(Child ~ Parents, data = Smoking))


boy.smokers
   -0.09698
```

1. H_0: $\pi_{smoker} - \pi_{nonsmoker} = 0$; H_a: $\pi_{smoker} - \pi_{nonsmoker} \neq 0$
2. Test statistic: $\hat{p}_{smoker} - \hat{p}_{nonsmoker} = -0.097$ (the difference in the conditional sample proportions)
3. We simulate a world in which $\pi_{smoker} - \pi_{nonsmoker} = 0$:

Example5.3

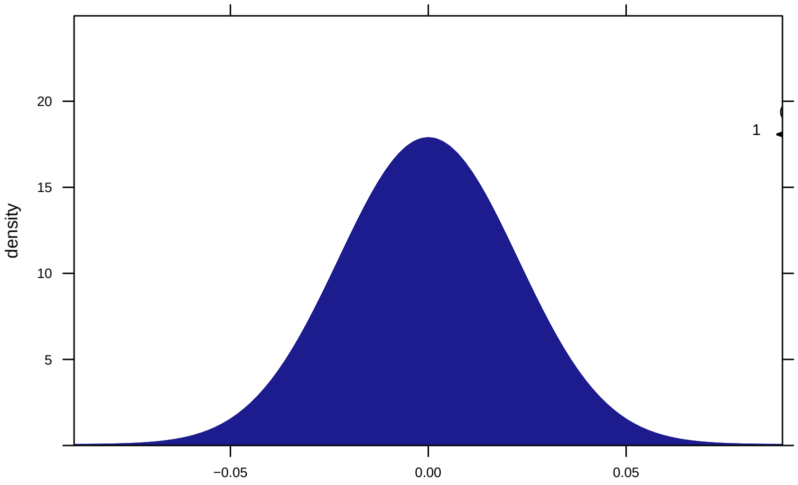
```
simulation.smoke <- do(5000) * diff(prop(shuffle(Child) ~ Parents, data = Smoking))
head(simulation.smoke, 3)


  boy.smokers
1    -0.01303
2     0.00335
3     0.02587

dotPlot(~boy.smokers, data = simulation.smoke)
```



```
favstats(~boy.smokers, data = simulation.smoke)

      min        Q1      median       Q3      max       mean       sd     n missing
 -0.07036 -0.01508 -0.0007449 0.01564 0.08525 0.0002293 0.02238 5000       0

prop(~(boy.smokers <= -0.097 | boy.smokers >= 0.097), data = simulation.smoke)

TRUE
   0
```

4. Normal approximation (using simulated standard deviation):

```
sd <- sd(~boy.smokers, data = simulation.smoke)
2 * xpnorm(0.097, 0, sd, lower.tail = FALSE)  # 2 times because two-sided


If X ~ N(0,0.022375248845317), then

P(X <= 0.097) = P(Z <= 4.335) = 1
P(X >  0.097) = P(Z >  4.335) = 0
[1] 1.457e-05
```



5. Approximate test for difference in proportions:

```
prop.test(Child ~ Parents, data = Smoking)


2-sample test for equality of proportions with continuity correction

data:  t(table_from_formula)
X-squared = 18.08, df = 1, p-value = 2.122e-05
alternative hypothesis: two.sided
95 percent confidence interval:
 0.05182 0.14214
sample estimates:
prop 1 prop 2
0.5483 0.4513
```

Estimation

Figure5.13

```
confint(prop.test(Child ~ Parents, data = Smoking))


 prop 1  prop 2   lower   upper   level
0.54831 0.45133 0.05182 0.14214 0.95000
```

Figure5.14

```
confint(prop.test(Child ~ Parents, data = Smoking, conf.level = 0.99))


 prop 1  prop 2   lower   upper   level
0.54831 0.45133 0.03795 0.15600 0.99000
```

Formulas

```
prop(Child ~ Parents, data = Smoking)


boy.nonsmokers    boy.smokers
        0.5483         0.4513


p.1 <- 0.548
p.2 <- 0.451
p.hat <- prop(~Child, data = Smoking)
p.hat  # pooled prop of success


   boy
0.5352


n.1 <- 565
n.2 <- 3602
```

```
z <- (p.1 - p.2)/sqrt((p.hat * (1 - p.hat) * (1/n.1 + 1/n.2)))
z


  boy
4.298
```

```
SE <- sqrt(p.1 * (1 - p.1)/n.1 + p.2 * (1 - p.2)/n.2)
SE


[1] 0.02252
```

```
MoE <- 2 * SE
MoE


[1] 0.04504
```

Exploration 5.3: Donating Blood

Exploration5.3.2

```
sample(Blood, 5)


     Year Response orig.ids
1361 2002  did.not     1361
416  2004  donated      416
1487 2002  did.not     1487
2405 2004  did.not     2405
20   2002  donated       20


tally(Response ~ Year, data = Blood, format = "count", margin = TRUE)


         Year
```

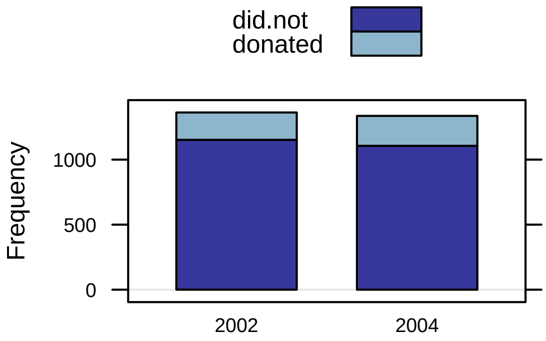
```
Response  2002 2004
  did.not 1152 1106
  donated  210  230
  Total   1362 1336
```

Exploration5.3.3

```
tally(Response ~ Year, data = Blood)


         Year
Response    2002   2004
  did.not 0.8458 0.8278
  donated 0.1542 0.1722


diff(prop(Response ~ Year, level = "donated", data = Blood))


donated.2004
     0.01797
```

Exploration5.3.4

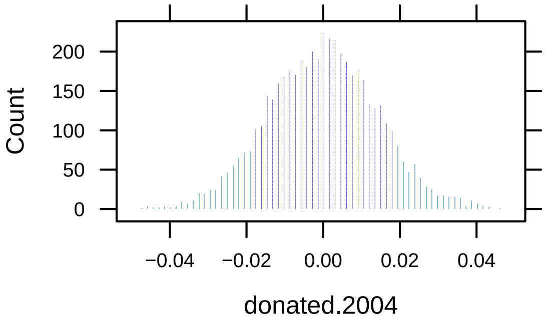
```
bargraph(~Year, groups = Response, data = Blood, stack = TRUE, auto.key = TRUE)
```



1. H_0: $\pi_{2004} - \pi_{2002} = 0$; H_a: $\pi_{2004} - \pi_{2002} \neq 0$
2. Test statistic: $\hat{p}_{2004} - \hat{p}_{2002} = 0.0180$ (the difference in the conditional sample proportions)
3. We simulate a world in which $\pi_{2004} - \pi_{2002} = 0$:

Exploration5.3.6

```
simulation.blood <-
  do(5000) * diff(prop(shuffle(Response) ~ Year, level = "donated", data = Blood))
head(simulation.blood, 3)

  donated.2004
1     0.025384
2     0.007592
3    -0.004270

dotPlot(~ donated.2004, data = simulation.blood,
        groups = (donated.2004 <= -0.018 | donated.2004 >= 0.018), width = 0.0001)
```

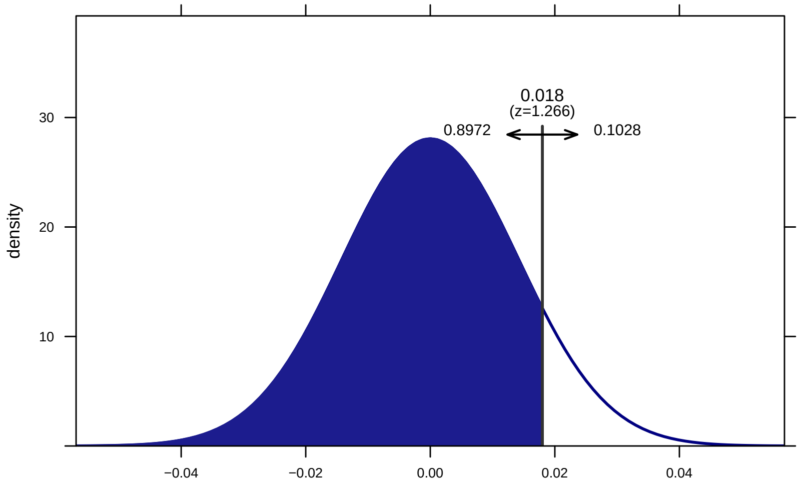


```
favstats(~donated.2004, data = simulation.blood)


      min       Q1    median        Q3     max       mean      sd    n missing
 -0.04727 -0.0102 0.0001781 0.009074 0.04614 -0.0001069 0.01422 5000       0

prop(~(donated.2004 <= -0.018 | donated.2004 >= 0.018), data = simulation.blood)

 TRUE
0.186
```

4. Normal approximation (using simulated standard deviation):

Exploration5.3.8

```
sd <- sd(~donated.2004, data = simulation.blood)
2 * xpnorm(0.018, 0, sd, lower.tail = FALSE)  # 2 times because two-sided


If X ~ N(0,0.0142186374869083), then

P(X <= 0.018) = P(Z <= 1.266) = 0.8972
P(X >  0.018) = P(Z >  1.266) = 0.1028
[1] 0.2055
```



5. Approximate test for difference in proportions:

```
prop.test(Response ~ Year, data = Blood)


2-sample test for equality of proportions with continuity correction

data:  t(table_from_formula)
X-squared = 1.467, df = 1, p-value = 0.2258
alternative hypothesis: two.sided
95 percent confidence interval:
 -0.01066  0.04660
sample estimates:
prop 1 prop 2
0.8458 0.8278
```

Exploration5.3.10

```
confint(prop.test(Response ~ Year, data = Blood))


  prop 1   prop 2    lower    upper    level
 0.84581  0.82784 -0.01066  0.04660  0.95000
```

Exploration5.3.11

```
success <- c(230, 210)
n <- c(1336, 1362)
prop.test(success, n)



2-sample test for equality of proportions with continuity correction

data:  x and n
X-squared = 1.467, df = 1, p-value = 0.2258
alternative hypothesis: two.sided
95 percent confidence interval:
 -0.01066  0.04660
sample estimates:
prop 1 prop 2
0.1722 0.1542
```

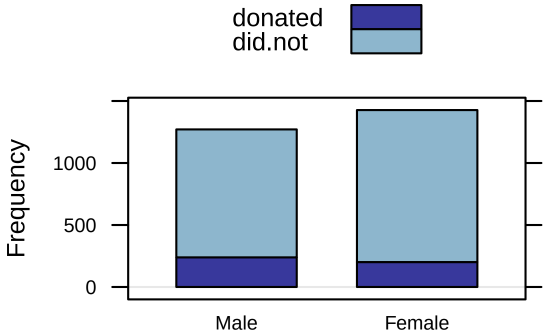
Exploration5.3.15

```
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")
   )
```

```
tally(~Response + Sex, data = Blood2, margin = TRUE)


         Sex
Response  Male Female Total
  donated  239    201   440
  did.not 1032   1226  2258
  Total   1271   1427  2698
```

```
tally(Response ~ Sex, data = Blood2)


         Sex
Response    Male Female
  donated 0.1880 0.1409
  did.not 0.8120 0.8591


diff(prop(Response ~ Sex, data = Blood2))


donated.Female
      -0.04719
```

```
bargraph(~Sex, data = Blood2, groups = Response, stack = TRUE, auto.key = TRUE)
```

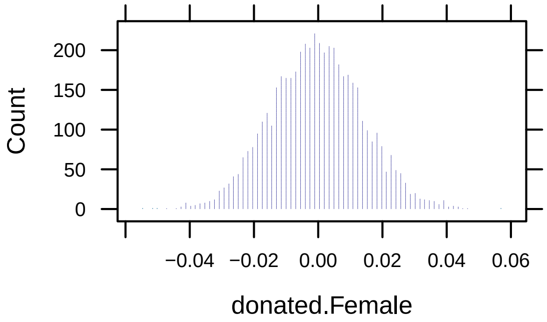


1. H_0: $\pi_{female} - \pi_{male} = 0$; H_a: $\pi_{female} - \pi_{male} \neq 0$

2. Test statistic: $\hat{p}_{female} - \hat{p}_{male} = -0.0472$ (the difference in the conditional sample proportions)

3. We simulate a world in which $\pi_{female} - \pi_{male} = 0$:

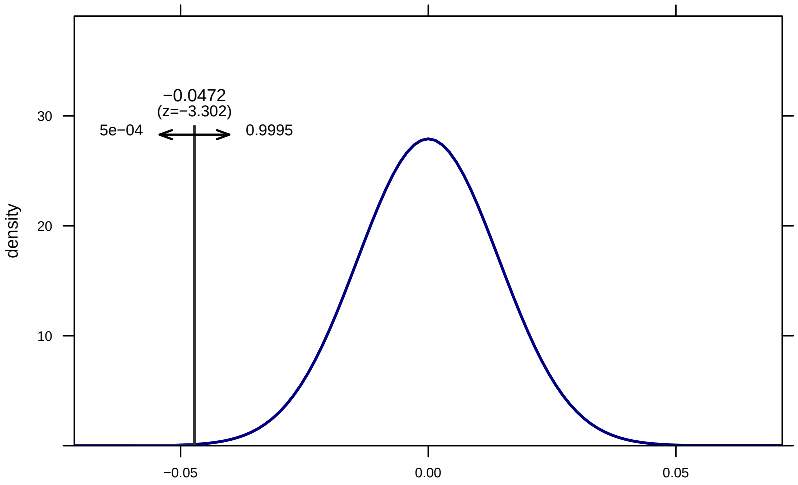
```
simulation.blood2 <- do(5000) * diff(prop(shuffle(Response) ~ Sex, data = Blood2))
head(simulation.blood2, 3)


  donated.Female
1      -0.017435
2       0.021241
3      -0.004047


dotPlot(~ donated.Female, data = simulation.blood2,
        groups = (donated.Female <= -0.0472 | donated.Female >= 0.0472), width = 0.0001)
```



```
favstats(~donated.Female, data = simulation.blood2)

      min        Q1    median       Q3     max       mean      sd    n missing
 -0.05462 -0.009997 -0.001072 0.009341 0.05694 -0.0004661 0.01429 5000       0

prop(~(donated.Female <= -0.0472 | donated.Female >= 0.0472), data = simulation.blood2)

 TRUE
8e-04
```

4. Normal approximation (using simulated standard deviation):

```
sd <- sd(~donated.Female, data = simulation.blood2)
2 * xpnorm(-0.0472, 0, sd, xlim = 0 + c(-5, 5) * sd)  # 2 times because two-sided


If X ~ N(0,0.0142927015599229), then

P(X <= -0.0472) = P(Z <= -3.302) = 5e-04
P(X >  -0.0472) = P(Z >  -3.302) = 0.9995
[1] 0.0009587
```



5. Approximate test for difference in proportions:

```
prop.test(Response ~ Sex, data = Blood2)


2-sample test for equality of proportions with continuity correction

data:  t(table_from_formula)
X-squared = 10.62, df = 1, p-value = 0.001117
alternative hypothesis: two.sided
95 percent confidence interval:
 0.01838 0.07599
sample estimates:
prop 1 prop 2
0.1880 0.1409
```

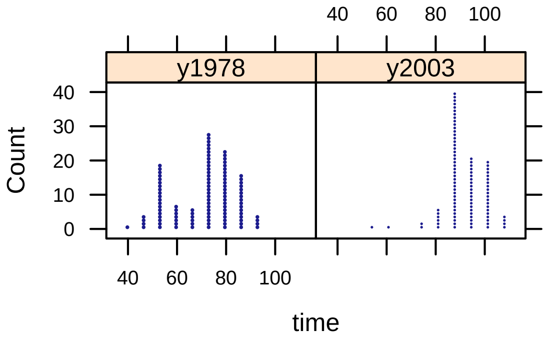
6 Comparing Two Means

6.1 Comparing Two Groups: Quantitative Response

Example 6.1: Geyser Eruptions

Figure6.1

```
head(OldFaithful, 3)

   year time
1 y1978   78
2 y1978   74
3 y1978   68

dotPlot(~time | year, data = OldFaithful)
```



Example6.1a

```
fivenum(~time, data = OldFaithful)

[1]  42  73  84  91 110

fivenum(time ~ year, data = OldFaithful)
```

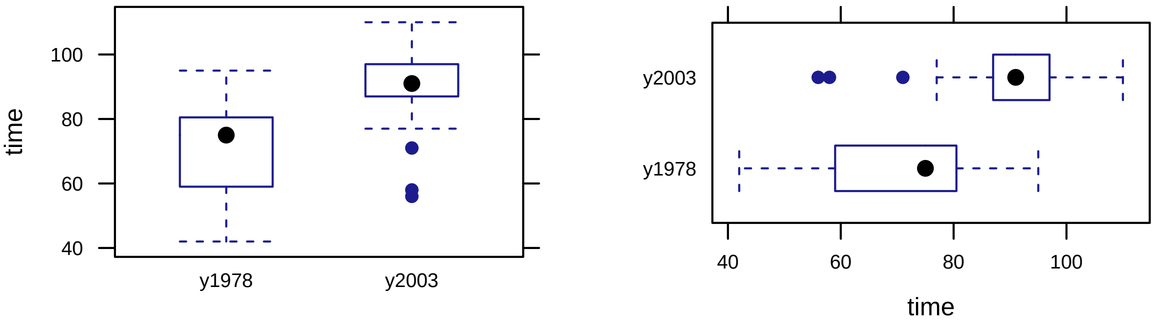
```
y19781 y19782 y19783 y19784 y19785 y20031 y20032 y20033 y20034 y20035
  42.0   59.0   75.0   80.5   95.0   56.0   87.0   91.0   97.0  110.0
```

Example6.1b

```
IQR(~time, data = OldFaithful)

[1] 18

IQR(~time | year, data = OldFaithful)

y1978 y2003
20.75 10.00
```

Figure6.2

```
bwplot(time ~ year, data = OldFaithful)
bwplot(year ~ time, data = OldFaithful, horizontal = TRUE)
```



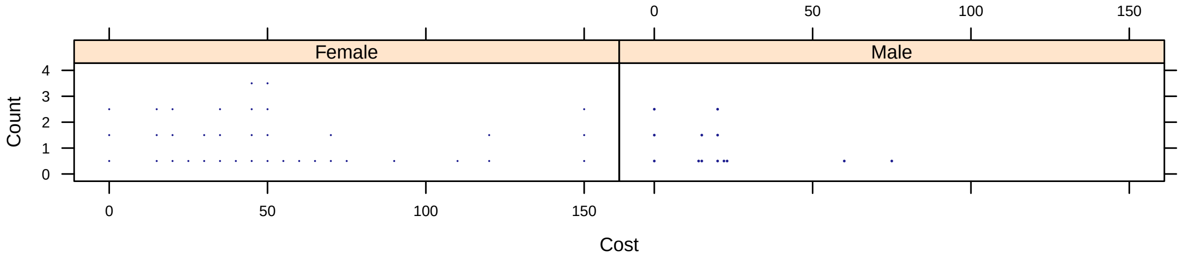
Exploration 6.1A: Haircut Prices

Exploration6.1A.1

```
head(Haircuts)

     Sex Cost
1 Female   50
2   Male   20
3 Female   60
4   Male   75
5 Female  150
6   Male   23
```

Exploration6.1A.4

```
dotPlot(~Cost | Sex, data = Haircuts, width = 1, cex = 0.1)
```



Exploration6.1A.8

```
favstats(~Cost | Sex, data = Haircuts)
```

```
  .group min Q1 median Q3 max  mean    sd  n missing
1 Female   0 25     45 70 150 54.05 41.61 37       0
2   Male   0 14     20 22  75 21.85 22.14 13       0
```

Exploration6.1A.10

```
diff(mean(Cost ~ Sex, data = Haircuts))
```

```
  Male
-32.21
```

Further Analyses

Exploration6.1A.14

```
median(Cost ~ Sex, data = Haircuts)
```

```
Female   Male
    45     20
```

Exploration6.1A.16

```
fivenum(~Cost, data = Haircuts)
```

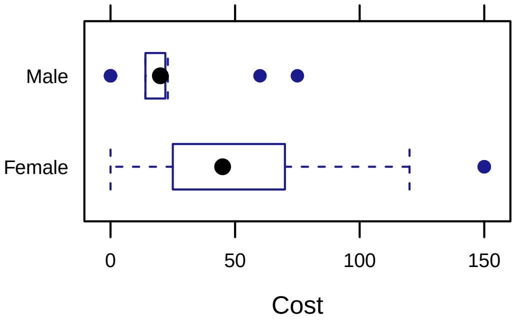
```
[1]   0  20  35  60 150
```

```
fivenum(~Cost | Sex, data = Haircuts)
```

```
Female1 Female2 Female3 Female4 Female5   Male1   Male2   Male3   Male4   Male5
      0      25      45      70     150       0      14      20      22      75
```

Exploration6.1A.17

```
bwplot(Sex ~ Cost, data = Haircuts, horizontal = TRUE)
```



Exploration6.1A.18

```
IQR(Cost ~ Sex, data = Haircuts)

Female   Male
    45      8
```

Exploration 6.1B: Cancer Pamplets

6.2 Comparing Two Means: Simulation-Based Approach

Example 6.2: Bicycling to Work

Table6.2

```
head(BikeTimes)

  Frame  Time
1 steel 115.8
2 steel 115.7
3 steel 108.7
4 steel 117.7
5 steel 112.6
6 steel 109.6
```

Figure6.3

```
bwplot(Frame ~ Time, data = BikeTimes, horizontal = TRUE)
```

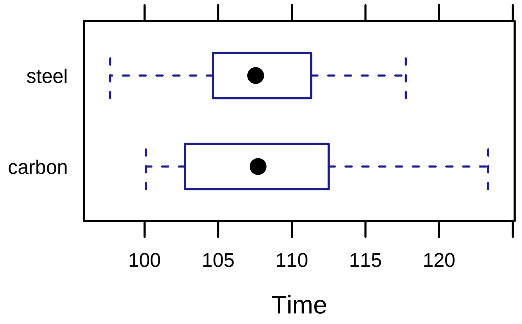


Table6.3

```
favstats(Time ~ Frame, data = BikeTimes)
```

```
  .group    min    Q1 median    Q3   max  mean    sd  n missing
1 carbon 100.08 102.8  107.7 112.5 123.3 108.3 6.248 26       0
2  steel  97.67 104.7  107.5 111.2 117.7 107.8 4.892 30       0
```

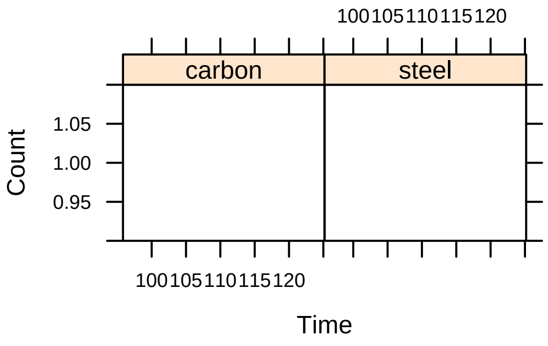
Figure6.4

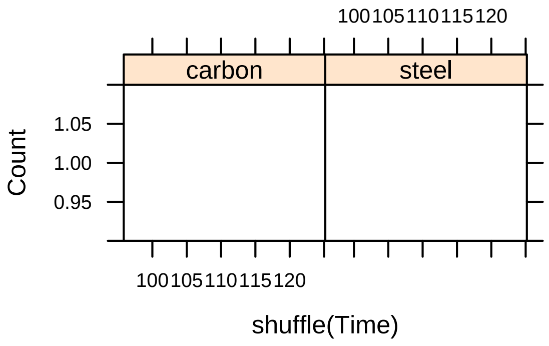
```
dotPlot(~Time | Frame, data = BikeTimes, width = 0.05, cex = 0.05, ylim = c(0.9, 1.1))
diff(mean(Time ~ Frame, data = BikeTimes))
```

```
  steel
-0.5347
```

```
dotPlot(~shuffle(Time) | Frame, data = BikeTimes, width = 0.05, cex = 0.05, ylim = c(0.9, 1.1))
diff(mean(shuffle(Time) ~ Frame, data = BikeTimes))
```

```
steel
1.215
```





1. H_0: $\mu_{carbon} - \mu_{steel} = 0$

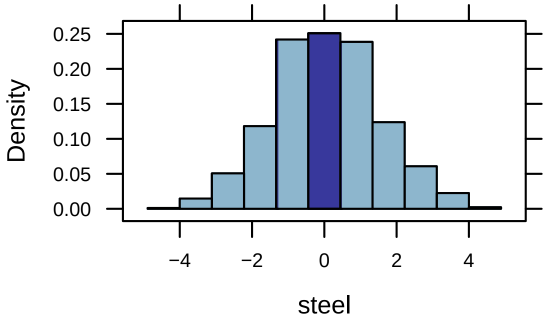
   H_a: $\mu_{carbon} - \mu_{steel} \neq 0$

   Test statistic: $\bar{x}_{carbon} - \bar{x}_{steel} = 0.53$ (the difference in the sample means)

2. We simulate a world in which $\mu_{carbon} - \mu_{steel} = 0$:

Figure6.7

```
simulation.bike <- do(1000) * diff(mean(shuffle(Time) ~ Frame, data = BikeTimes))

Loading required package:  parallel

head(simulation.bike, 3)

    steel
1  0.6104
2  0.9024
3 -0.2691

histogram(~ steel, data = simulation.bike,
        groups = (steel <= -0.53 | steel >= 0.53))
```



3. Strength of evidence:

Figure6.8

```
favstats(~steel, data = simulation.bike)

    min      Q1  median    Q3   max    mean    sd    n missing
 -4.553 -0.9047 0.05581 1.037 4.335 0.08303 1.454 1000       0

prop(~(steel <= -0.53 | steel >= 0.53), data = simulation.bike)

 TRUE
0.738
```

Estimating a confidence interval

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

Example6.2

```
diff <- -diff(mean(Time ~ Frame, data = BikeTimes))  # note the negative sign
sd <- sd(~steel, data = simulation.bike)
diff - 2 * sd  # lower limit of 95% CI
```

```
 steel
-2.372


diff + 2 * sd  # upper limit of 95% CI


steel
3.442
```

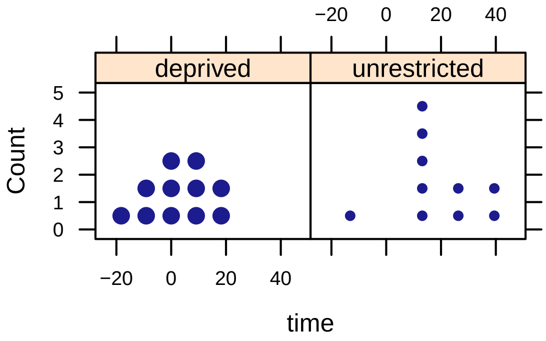
Exploration 6.2: Lingering Effects of Sleep Deprivation

Exploration6.2.2

```
head(Sleep)


         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
```

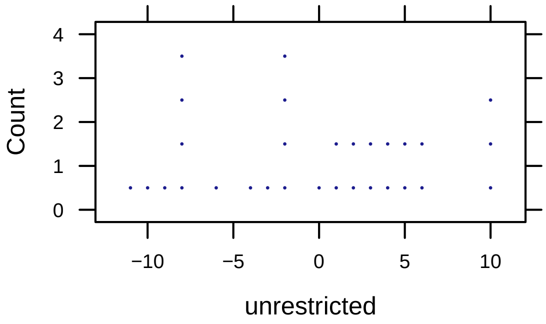
Exploration6.2.5

```
dotPlot(~time | sleep, data = Sleep, cex = 0.5)
favstats(time ~ sleep, data = Sleep)


        .group   min    Q1 median    Q3  max  mean    sd  n missing
1     deprived -14.7 -4.25   4.50  9.80 21.8  3.90 12.17 11       0
2 unrestricted  -7.0 12.22  16.55 29.18 45.6 19.82 14.73 10       0


diff(mean(time ~ sleep, data = Sleep))


unrestricted
       15.92
```



Exploration6.2.9

```
diff(mean(shuffle(time) ~ sleep, data = Sleep))


unrestricted
      -15.1


sample <- do(30) * diff(mean(shuffle(time) ~ sleep, data = Sleep))
head(sample, 3)


  unrestricted
1       -1.720
2       -7.715
3        3.988


dotPlot(~unrestricted, data = sample, width = 1, cex = 0.1)
```



1. H_0: $\mu_{unrestricted} - \mu_{deprived} = 0$

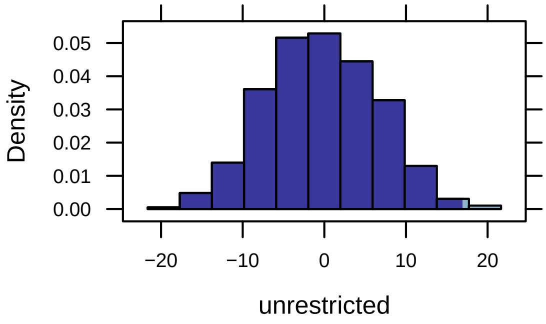
   H_a: $\mu_{unrestricted} - \mu_{deprived} > 0$

   Test statistic: $\bar{x}_{unrestricted} - \bar{x}_{deprived} = 15.92$ (the difference in the sample means)

2. We simulate a world in which $\mu_{unrestricted} - \mu_{deprived} = 0$:

Exploration6.2.10

```
simulation.sleep <- do(1000) * diff(mean(shuffle(time) ~ sleep, data = Sleep))
head(simulation.sleep, 3)


  unrestricted
1      -1.5291
2       0.7045
3      -5.5000


histogram(~ unrestricted, data = simulation.sleep,
        groups = (unrestricted >= 15.92))
```



3. Strength of evidence:

Exploration6.2.10b

```
favstats(˜unrestricted, data = simulation.sleep)

    min     Q1  median    Q3   max    mean    sd    n missing
 -19.97 -5.027 -0.4886 4.623 19.38 -0.3325 6.829 1000       0

prop(˜(unrestricted >= 15.92), data = simulation.sleep)

 TRUE
0.006
```

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

Exploration6.2.13

```
diff <- diff(mean(time ˜ sleep, data = Sleep))
sd <- sd(˜unrestricted, data = simulation.sleep)
diff - 2 * sd  # lower limit of 95% CI

unrestricted
       2.263

diff + 2 * sd  # upper limit of 95% CI

unrestricted
       29.58
```

Another statistic

Exploration6.2.16a

```
median(time ˜ sleep, data = Sleep)

    deprived unrestricted
        4.50        16.55
```

```
diff(median(time ~ sleep, data = Sleep))

unrestricted
       12.05
```

1. H_0: median$_{unrestricted}$ - median$_{deprived}$ = 0

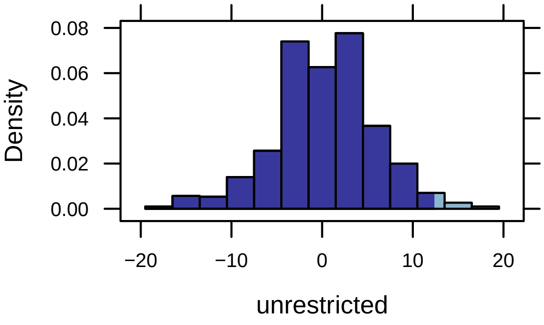
   H_a: median$_{unrestricted}$ - median$_{deprived}$ > 0

   Test statistic: median$_{unrestricted}$ - median$_{deprived}$ = 12.05 (the difference in the sample medians)

2. We simulate a world in which median$_{unrestricted}$ - median$_{deprived}$ = 0:

Exploration6.2.16b

```
simulation.med <- do(1000) * diff(median(shuffle(time) ~ sleep, data = Sleep))
head(simulation.med, 3)

  unrestricted
1         2.50
2       -15.15
3         3.95

histogram(~ unrestricted, data = simulation.med,
        groups = (unrestricted >= 12.05),
        width = 3)
```



3. Strength of evidence:

Exploration6.2.16c

```
favstats(~unrestricted, data = simulation.med)

 min   Q1 median   Q3  max   mean    sd    n missing
 -19 -2.8   -0.5 3.45 17.8 0.1648 5.591 1000       0

prop(~(unrestricted >= 12.05), data = simulation.med)

 TRUE
0.018
```

6.3 Comparing Two Means: Theory-Based Approach

Example 6.3: Breastfeeding and Intelligence

Table6.4

```
head(BreastFeedIntell)
```

```
    Feeding    GCI
1 Breastfed 126.70
2 Breastfed 124.69
3 Breastfed  99.79
4 Breastfed 104.97
5 Breastfed  97.25
6 Breastfed 131.28
```

```
favstats(GCI ~ Feeding, data = BreastFeedIntell)
```

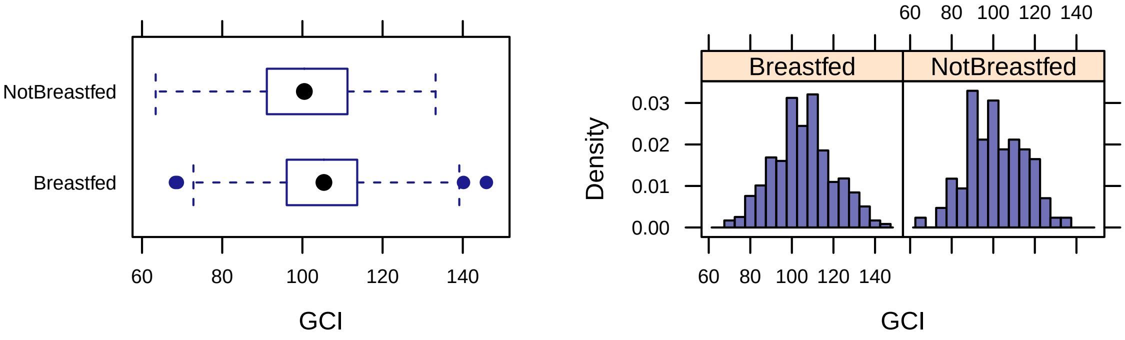
```
        .group   min    Q1 median    Q3   max  mean   sd   n missing
1    Breastfed 68.33 96.08  105.4 113.7 145.9 105.3 14.5 237       0
2 NotBreastfed 63.41 91.13  100.5 111.2 133.2 100.9 14.0  85       0
```

```
diff(mean(GCI ~ Feeding, data = BreastFeedIntell))
```

```
NotBreastfed
        -4.4
```

Figure6.10

```
bwplot(Feeding ~ GCI, horizontal = TRUE, data = BreastFeedIntell)
histogram(~GCI | Feeding, data = BreastFeedIntell, width = 5)
```



1. H_0: $\mu_{breastfed} - \mu_{not} = 0$

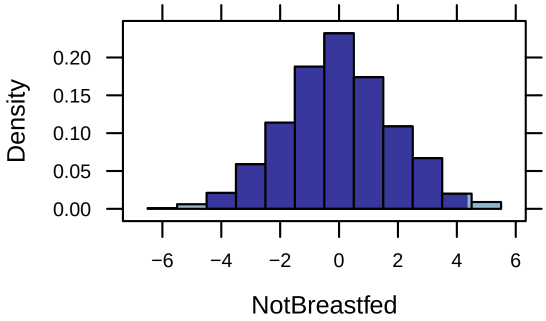
   H_a: $\mu_{breastfed} - \mu_{not} \neq 0$

   Test statistic: $\bar{x}_{breastfed} - \bar{x}_{not} = 4.40$ (the difference in the sample means)

2. We simulate a world in which $\mu_{breastfed} - \mu_{not} = 0$:

Figure6.11

```
simulation.GCI <- do(1000) * diff(mean(shuffle(GCI) ~ Feeding, data = BreastFeedIntell))
head(simulation.GCI, 3)

  NotBreastfed
1        0.481
2       -2.402
3        3.240

histogram(~ NotBreastfed, data = simulation.GCI,
        groups = (NotBreastfed <= -4.40 | NotBreastfed >= 4.40), width = 1)
```



3. Strength of evidence:

Figure6.12

```
favstats(~NotBreastfed, data = simulation.GCI)

    min     Q1    median     Q3    max       mean     sd     n missing
 -5.947 -1.228 -0.03834 1.184 5.156 -0.008255 1.838 1000       0

prop(~(NotBreastfed <= -4.4 | NotBreastfed >= 4.4), data = simulation.GCI)

 TRUE
0.018
```

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

Example6.3a

```
diff <- -diff(mean(GCI ~ Feeding, data = BreastFeedIntell))  # note the negative sign
sd <- sd(~NotBreastfed, data = simulation.GCI)
sd

[1] 1.838

diff - 2 * sd  # lower limit of 95% CI

NotBreastfed
      0.7242
```

```
diff + 2 * sd  # upper limit of 95% CI


NotBreastfed
       8.076
```

Figure6.13

```
t.test(GCI ~ Feeding, data = BreastFeedIntell)



Welch Two Sample t-test

data:  GCI by Feeding
t = 2.462, df = 153, p-value = 0.01491
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.8699 7.9302
sample estimates:
   mean in group Breastfed mean in group NotBreastfed
                     105.3                      100.9


stat(t.test(GCI ~ Feeding, data = BreastFeedIntell))


    t
2.462
```

Exploration 6.3: Close Friends

Exploration6.3.1

```
head(CloseFriends)


  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
```

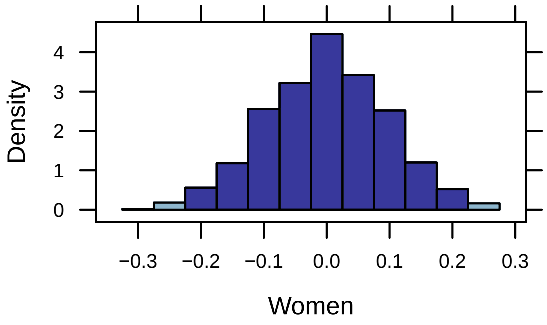
Exploration6.3.7

```
favstats(Friends ~ Sex, data = CloseFriends)

  .group min Q1 median Q3 max  mean    sd   n missing
1    Men   0  0      1  3   6 1.861 1.777 654       0
2  Women   0  1      2  3   6 2.089 1.760 813       0

diff(mean(Friends ~ Sex, data = CloseFriends))

 Women
0.2277
```

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 $\mu_{men} - \mu_{women} = 0$:

Exploration6.3.8

```
simulation.fri <- do(1000) * diff(mean(shuffle(Friends) ~ Sex, data = CloseFriends))
head(simulation.fri, 3)

     Women
1 0.100788
2 0.001461
3 0.053884

histogram(~ Women, data = simulation.fri,
        groups = (Women <= -0.228 | Women >= 0.228), width = 0.05)
```



3. Strength of evidence:

Exploration6.3.10

```
favstats(~Women, data = simulation.fri)

     min       Q1   median      Q3    max      mean      sd    n missing
 -0.3076 -0.06752 0.001461 0.06216 0.2691 -0.000561 0.09431 1000       0

prop(~(Women <= -0.228 | Women >= 0.228), data = simulation.fri)

 TRUE
0.018
```

Exploration6.3.13

```
t.test(Friends ~ Sex, data = CloseFriends)


Welch Two Sample t-test

data:  Friends by Sex
t = -2.45, df = 1393, p-value = 0.01442
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.41004 -0.04537
sample estimates:
  mean in group Men mean in group Women
              1.861               2.089


stat(t.test(Friends ~ Sex, data = CloseFriends))


    t
-2.45
```

Exploration6.3.17

```
pval(t.test(Friends ~ Sex, data = CloseFriends))


p.value
0.01442
```

Validity Conditions

Exploration6.3.20

```
confint(t.test(Friends ~ Sex, data = CloseFriends))


  mean in group Men mean in group Women               lower               upper
            1.86086             2.08856            -0.41004            -0.04537
              level
            0.95000
```

7 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)

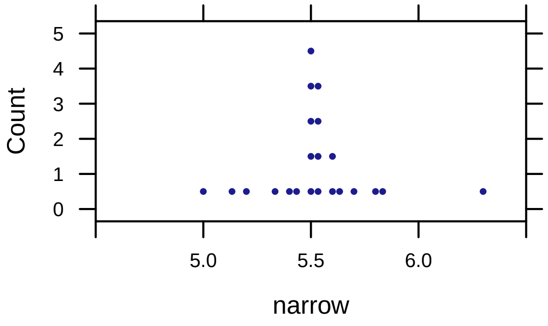
Table7.1

```
head(FirstBase, 10)
```

```
   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
```

Figure7.3

```
dotPlot(~narrow, data = FirstBase, nint = 40, xlim = c(4.5, 6.5), cex = 0.25)
dotPlot(~wide, data = FirstBase, nint = 40, xlim = c(4.5, 6.5), cex = 0.15)
```



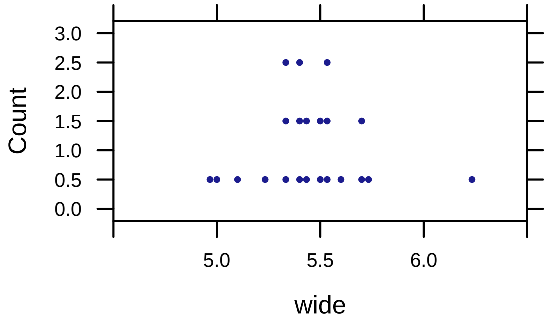
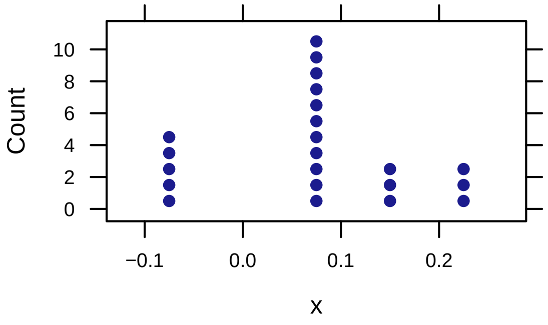


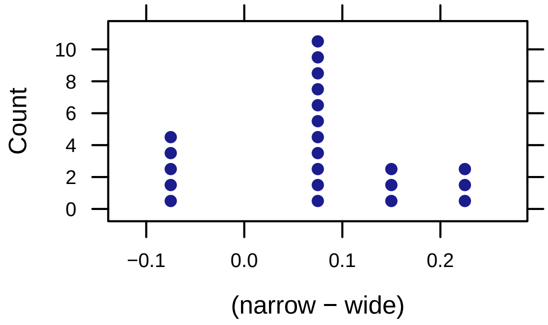
Table7.2

```
FirstBase$narrow - FirstBase$wide

 [1] -0.05 -0.05  0.10  0.10  0.15 -0.05  0.05  0.15  0.15  0.10  0.10  0.10 -0.10  0.05
[15]  0.10  0.05  0.20 -0.05  0.20  0.20  0.10  0.05

favstats(FirstBase$narrow - FirstBase$wide)

  min   Q1 median     Q3 max  mean     sd  n missing
 -0.1 0.05    0.1 0.1375 0.2 0.075 0.0883 22       0

favstats(~(narrow - wide), data = FirstBase)

  min   Q1 median     Q3 max  mean     sd  n missing
 -0.1 0.05    0.1 0.1375 0.2 0.075 0.0883 22       0
```

Figure7.4

```
dotPlot(FirstBase$narrow - FirstBase$wide)
dotPlot(~(narrow - wide), data = FirstBase)
```





We simulate a world in which $\mu_d = 0$:

Exploration 7.2: Exercise and Heart Rate

Exploration7.2.5

```
head(JJvsBicycle)
```

```
   JJ bicycle
1 118     118
2 146     124
3 134      92
4  94      80
5 146     111
6 114     112
```

Exploration7.2.7

```
favstats(~JJ, data = JJvsBicycle)
```

```
 min    Q1 median    Q3 max  mean    sd  n missing
  70 102.2    115 129.2 146 114.6 19.57 22       0
```

```
favstats(~bicycle, data = JJvsBicycle)
```

```
 min    Q1 median    Q3 max  mean    sd  n missing
  70 87.25   97.5 121.8 143 102.7 20.66 22       0
```

```
mean(JJvsBicycle$JJ - JJvsBicycle$bicycle)
```

```
[1] 11.95
```

1. H_0: $\mu_d = 0$
   H_a: $\mu_d \neq 0$
   Test statistic: $\bar{x}_d$ = (the mean difference in sample)
2. We simulate a world in which $\mu_d = 0$:
3. Strength of evidence:

Standarized statistic:

95% confidence interval using 2SD Method:

7.3 Theory-Based Approach to Analyzing Data from Paired Samples

Example 7.3: How Many M&Ms Would You Like?

Table7.4

```
head(BowlsMMs)
```

```
  Small Large
1    33    41
```

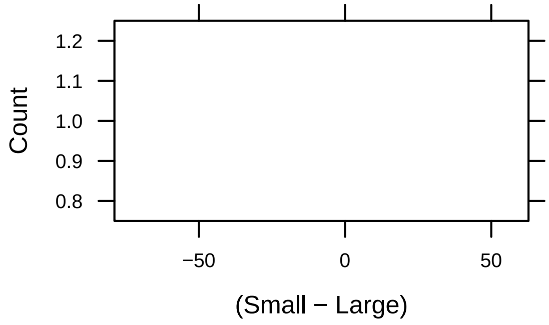
```
2    24    92
3    35    61
4    24    19
5    40    21
6    33    35
```

Table7.5

```
favstats(~Small, data = BowlsMMs)


 min Q1 median Q3 max  mean   sd  n missing
  24 26     34 40  88 38.59 16.9 17       0


favstats(~Large, data = BowlsMMs)


 min Q1 median Q3 max  mean    sd  n missing
  11 33     42 62 104 49.47 27.21 17       0


favstats(BowlsMMs$Small - BowlsMMs$Large)


 min  Q1 median Q3 max   mean   sd  n missing
 -69 -28     -8 14  54 -10.88 36.3 17       0
```

Figure7.9

```
dotPlot(~(Small - Large), data = BowlsMMs, nint = 100, ylim = c(0.75, 1.25), cex = 0.05)
```



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 $\mu_d = 0$:

3. Strength of evidence:

Theory-based approach

Figure7.12

```
t.test(BowlsMMs$Small, BowlsMMs$Large, paired = TRUE, alt = "less")
```

```
Paired t-test

data:  x and BowlsMMs$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 4.489
sample estimates:
mean of the differences
                 -10.88
```

Exploration 7.3: comparing Auction Formats

Exploration7.3.1

```
head(Auction)
```

```
  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
```

Exploration7.3.5a

```
summary(Auction)
```

```
     dutch             FP
 Min.   : 0.15   Min.   : 0.10
 1st Qu.: 2.00   1st Qu.: 1.19
 Median : 3.00   Median : 2.27
 Mean   : 5.16   Mean   : 4.78
 3rd Qu.: 7.00   3rd Qu.: 6.05
 Max.   :26.00   Max.   :27.00
```

```
favstats(Auction$dutch - Auction$FP)
```

```
   min Q1 median  Q3 max   mean     sd  n missing
 -1.25  0   0.25 0.5 2.4 0.3835 0.6752 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 $\mu_d = 0$:

3. Strength of evidence:
4. t-test for paired samples (theory-based approach):

Exploration7.3.7

```
t.test(Auction$dutch, Auction$FP, paired = TRUE)


Paired t-test

data:  x and Auction$FP
t = 5.328, df = 87, p-value = 7.692e-07
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.2405 0.5266
sample estimates:
mean of the differences
                 0.3835

t.test(~(dutch - FP), data = Auction)


One Sample t-test

data:  data$(dutch - FP)
t = 5.328, df = 87, p-value = 7.692e-07
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 0.2405 0.5266
sample estimates:
mean of x
   0.3835
```

95% confidence interval using the t-test:

Exploration7.3.8

```
confint(t.test(Auction$dutch, Auction$FP, paired = TRUE))


mean of the differences                   lower                   upper
                 0.3835                  0.2405                  0.5266
                  level
                 0.9500
```

Index