Chapter 13

Hypothesis Tests

In this chapter we'll cover 1 and 2 sample null hypothesis tests: like the t-test, correlation test, and chi-square test:

```r
library(yarrr) # Load yarrr to get the pirates data

# 1 sample t-test
# Are pirate ages different than 30 on average?
t.test(x = pirates$age,
       mu = 30)

# 2 sample t-test
# Do females and males have different numbers of  tattoos?
sex.ttest <- t.test(formula = tattoos ~ sex,
                    data = pirates,
                    subset = sex %in% c("male", "female"))
sex.ttest # Print result

## Access specific values from test
sex.ttest$statistic
sex.ttest$p.value
sex.ttest$conf.int

# Correlation test
# Is there a relationship between age and height?
 cor.test(formula = ~ age + height,
          data = pirates)

# Chi-Square test
# Is there a relationship between college and favorite pirate?
chisq.test(x = pirates$college,
           y = pirates$favorite.pirate)
```

Do we get more treasure from chests buried in sand or at the bottom of the ocean? Is there a relationship between the number of scars a pirate has and how much grogg he can drink? Are pirates with body piercings more likely to wear bandannas than those without body piercings? Glad you asked, in this chapter, we'll answer these questions using 1 and 2 sample frequentist hypothesis tests.

As this is a Pirate's Guide to R, and not a Pirate's Guide to Statistics, we won't cover all the theoretical background behind frequentist null hypothesis tests (like t-tests) in much detail. However, it's important to

Figure 13.1: Sadly, this still counts as just one tattoo.

cover three main concepts: Descriptive statistics, Test statistics, and p-values. To do this, let's talk about body piercings.

13.1 A short introduction to hypothesis tests

As you may know, pirates are quite fond of body piercings. Both as a fashion statement, and as a handy place to hang their laundry. Now, there is a stereotype that European pirates have more body piercings than American pirates. But is this true? To answer this, I conducted a survey where I asked 10 American and 10 European pirates how many body piercings they had. The results are below, and a Pirateplot of the data is in Figure **??**:

```
# Body piercing data
american.bp <- c(3, 5, 2, 1, 4, 4, 6, 3, 5, 4)
european.bp <- c(6, 5, 7, 7, 6, 3, 4, 6, 5, 4)

# Store data in a dataframe
bp.survey <- data.frame("bp" = c(american.bp, european.bp),
                        "group" = rep(c("American", "European"), each = 10),
                         stringsAsFactors = FALSE)
```

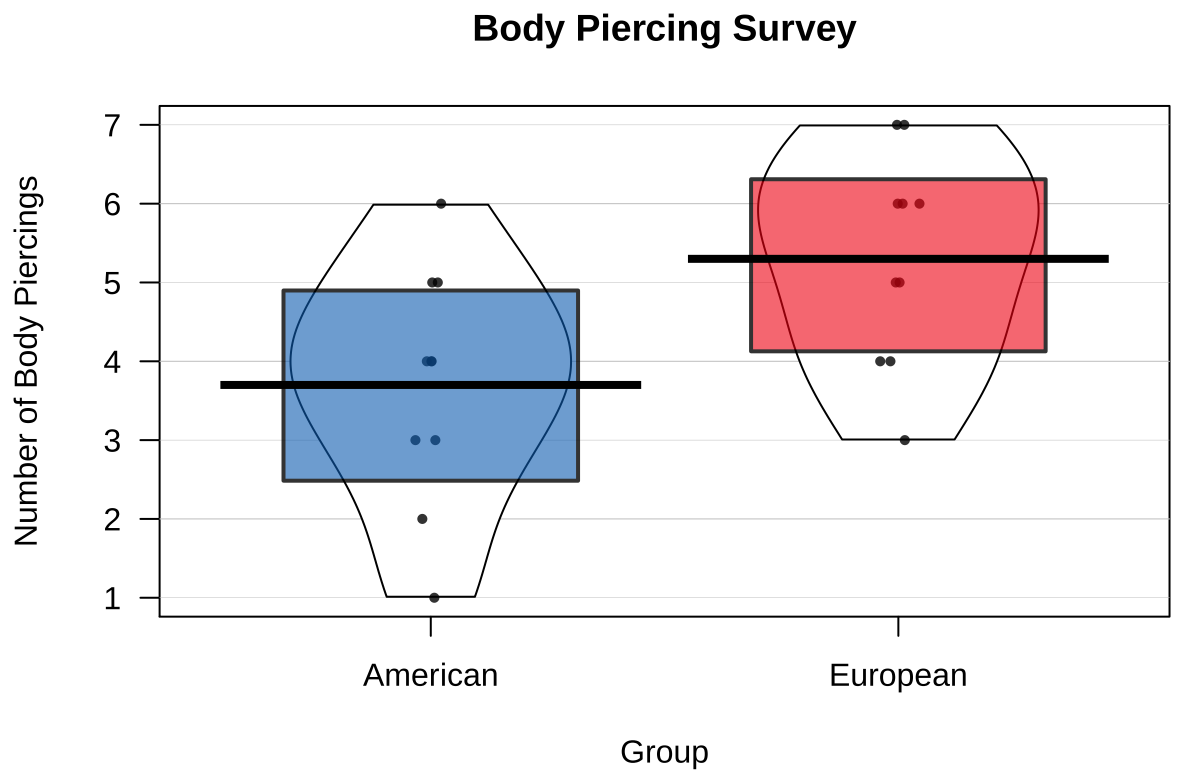


Figure 13.2: A Pirateplot of the body piercing data.

```r
yarrr::pirateplot(bp ~ group,
                  data = bp.survey,
                  main = "Body Piercing Survey",
                  ylab = "Number of Body Piercings",
                  xlab = "Group",
                  theme = 2, point.o = .8, cap.beans = TRUE)
```

13.1.1 Null v Alternative Hypothesis

In null hypothesis tests, you always start with a *null hypothesis*. The specific null hypothesis you choose will depend on the type of question you are asking, but in general, the null hypothesis states that *nothing is going on and everything is the same*. For example, in our body piercing study, our null hypothesis is that American and European pirates have the *same* number of body piercings on average.

The *alternative* hypothesis is the opposite of the null hypothesis. In this case, our alternative hypothesis is that American and European pirates do *not* have the same number of piercings on average. We can have different types of alternative hypotheses depending on how specific we want to be about our prediction. We can make a *1-sided* (also called 1-tailed) hypothesis, by predicting the *direction* of the difference between American and European pirates. For example, our alternative hypothesis could be that European pirates have *more* piercings on average than American pirates.

Alternatively, we could make a *2-sided* (also called 2-tailed) alternative hypothesis that American and European pirates simply differ in their average number of piercings, without stating which group has more piercings than the other.

Once we've stated our null and alternative hypotheses, we collect data and then calculate *descriptive* statistics.

13.1.2 Descriptive statistics

Descriptive statistics (also called sample statistics) describe samples of data. For example, a mean, median, or standard deviation of a dataset is a descriptive statistic of that dataset. Let's calculate some descriptive statistics on our body piercing survey American and European pirates using the `summary()` function:

```
# Pring descriptive statistics of the piercing data
summary(american.bp)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     1.0     3.0     4.0     3.7     4.8     6.0
summary(european.bp)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     3.0     4.2     5.5     5.3     6.0     7.0
```

Well, it looks like our sample of 10 American pirates had 3.7 body piercings on average, while our sample of 10 European pirates had 5.3 piercings on average. But is this difference large or small? Are we justified in concluding that American and European pirates *in general* differ in how many body piercings they have? To answer this, we need to calculate a *test statistic*

13.1.3 Test Statistics

An test statistic compares descriptive statistics, and determines how different they are. The formula you use to calculate a test statistics depends the type of test you are conducting, which depends on many factors, from the scale of the data (i.e.; is it nominal or interval?), to how it was collected (i.e.; was the data collected from the same person over time or were they all different people?), to how its distributed (i.e.; is it bell-shaped or highly skewed?).

For now, I can tell you that the type of data we are analyzing calls for a two-sample T-test. This test will take the descriptive statistics from our study, and return a test-statistic we can then use to make a decision about whether American and European pirates really differ. To calculate a test statistic from a two-sample t-test, we can use the `t.test()` function in R. Don't worry if it's confusing for now, we'll go through it in detail shortly.

```
# Conduct a two-sided t-test comparing the vectors american.bp and european.bp
#  and save the results in an object called bp.test
bp.test <- t.test(x = american.bp,
                  y = european.bp,
                  alternative = "two.sided")

# Print the main results
bp.test
##
##  Welch Two Sample t-test
##
## data:  american.bp and european.bp
## t = -3, df = 20, p-value = 0.02
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.93 -0.27
## sample estimates:
## mean of x mean of y
##       3.7       5.3
```

Figure 13.3: p-values are like bullshit detectors against the null hypothesis. The smaller the p-value, the more likely it is that the null-hypothesis (the idea that the groups are the same) is bullshit.

It looks like our test-statistic is -2.52. If there was really no difference between the groups of pirates, we would expect a test statistic close to 0. Because test-statistic is -2.52, this makes us think that there really is a difference. However, in order to make our decision, we need to get the *p-value* from the test.

13.1.4 p-value

The p-value is a probability that reflects how consistent the test statistic is *with the hypothesis that the groups are actually the same*. Or more formally, a p-value can be interpreted as follows:

13.1.4.1 Definition of a p-value: Assuming that there the null hypothesis is true (i.e.; that there is no difference between the groups), what is the probability that we would have gotten a test statistic as far away from 0 as the one we actually got?

For this problem, we can access the p-value as follows:

```
# What is the p-value from our t-test?
bp.test$p.value
## [1] 0.021
```

The p-value we got was 0.02, this means that, assuming the two populations of American and European pirates have the same number of body piercings on average, the probability that we would obtain a test statistic as large as -2.52 is 2.1% . This is very small, but is it small enough to decide that the null hypothesis is not true? It's hard to say and there is no definitive answer. However, most pirates use a decision threshold of $p < 0.05$ to determine if we should reject the null hypothesis or not. In other words, if you obtain a p-value less than 0.05, then you reject the null hypothesis. Because our p-value of 0.02 is less than 0.05, we would reject the null hypothesis and conclude that the two populations are *not* be the same.

13.1.4.2 p-values are bullshit detectors against the null hypothesis

P-values sounds complicated – because they are (In fact, most psychology PhDs get the definition wrong). It's very easy to get confused and not know what they are or how to use them. But let me help by putting it another way: a p-value is like a bullshit detector *against* the null hypothesis that goes off when the p-value is too small. If a p-value is too small, the bullshit detector goes off and says "Bullshit! There's no way you would get data like that if the groups were the same!" If a p-value is not too small, the bullshit alarm stays silent, and we conclude that we cannot reject the null hypothesis.

Figure 13.4: Despite what you may see in movies, men cannot get pregnant. And despite what you may want to believe, p-values do not tell you the probability that the null hypothesis is true!

13.1.4.3 How small of a p-value is too small?

Traditionally a p-value of 0.05 (or sometimes 0.01) is used to determine 'statistical significance.' In other words, if a p-value is less than .05, most researchers then conclude that the null hypothesis is false. However, .05 is not a magical number. Anyone who really believes that a p-value of .06 is *much* less significant than a p-value of 0.04 has been sniffing too much glue. However, in order to be consistent with tradition, I will adopt this threshold for the remainder of this chapter. That said, let me reiterate that a p-value threshold of 0.05 is just as arbitrary as a p-value of 0.09, 0.06, or 0.12156325234.

13.1.4.4 Does the p-value tell us the probability that the null hypothesis is true?

No!!! The p-value does not tell you the probability that the null hypothesis is true. In other words, if you calculate a p-value of .04, this does not mean that the probability that the null hypothesis is true is 4%. Rather, it means that *if the null hypothesis was true*, the probability of obtaining the result you got is 4%. Now, this does indeed set off our bullshit detector, but again, it does not mean that the probability that the null hypothesis is true is 4%.

Let me convince you of this with a short example. Imagine that you and your partner have been trying to have a baby for the past year. One day, your partner calls you and says "Guess what! I took a pregnancy test and it came back positive!! I'm pregnant!!" So, given the positive pregnancy test, what is the probability that your partner is really pregnant?

Now, imagine that the pregnancy test your partner took gives incorrect results in 1% of cases. In other words, if you *are* pregnant, there is a 1% chance that the test will make a mistake and say that you are *not* pregnant. If you really are *not* pregnant, there is a 1% change that the test make a mistake and say you *are* pregnant.

Ok, so in this case, the null hypothesis here is that your partner is *not* pregnant, and the alternative hypothesis is that they *are* pregnant. Now, if the null hypothesis is true, then the probability that they would have gotten an (incorrect) positive test result is just 1%. Does this mean that the probability that your partner is *not* pregnant is only 1%.

No. Your partner is a man. The probability that the null hypothesis is true (i.e. that he is not pregnant), is 100%, not 1%. Your stupid boyfriend doesn't understand basic biology and decided to buy an expensive pregnancy test anyway.

This is an extreme example of course – in most tests that you do, there will be some positive probability that the null hypothesis is false. However, in order to reasonably calculate an accurate probability that the null hypothesis is true after collecting data, you *must* take into account the *prior* probability that the null hypothesis was true before you collected data. The method we use to do this is with Bayesian statistics. We'll go over Bayesian statistics in a later chapter.

13.2 Hypothesis test objects: `htest`

R stores hypothesis tests in special object classes called `htest`. `htest` objects contain all the major results from a hypothesis test, from the test statistic (e.g.; a t-statistic for a t-test, or a correlation coefficient for a correlation test), to the p-value, to a confidence interval. To show you how this works, let's create an `htest` object called `height.htest` containing the results from a two-sample t-test comparing the heights of male and female pirates:

```
# T-test comparing male and female heights
#  stored in a new htest object called height.htest
height.htest <- t.test(formula = height ~ sex,
                       data = pirates,
                       subset = sex %in% c("male", "female"))
```

Once you've created an `htest` object, you can view a print-out of the main results by just evaluating the object name:

```
# Print main results from height.htest
height.htest
##
##  Welch Two Sample t-test
##
## data:  height by sex
## t = -20, df = 1000, p-value <2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -15 -13
## sample estimates:
## mean in group female   mean in group male
##                  163                  177
```

Just like in dataframes, you can also access specific elements of the `htest` object by using the `$` symbol. To see all the named elements in the object, run `names()`:

```
# Show me all the elements in the height.htest object
names(height.htest)
## [1] "statistic"   "parameter"   "p.value"     "conf.int"    "estimate"
## [6] "null.value"  "alternative" "method"      "data.name"
```

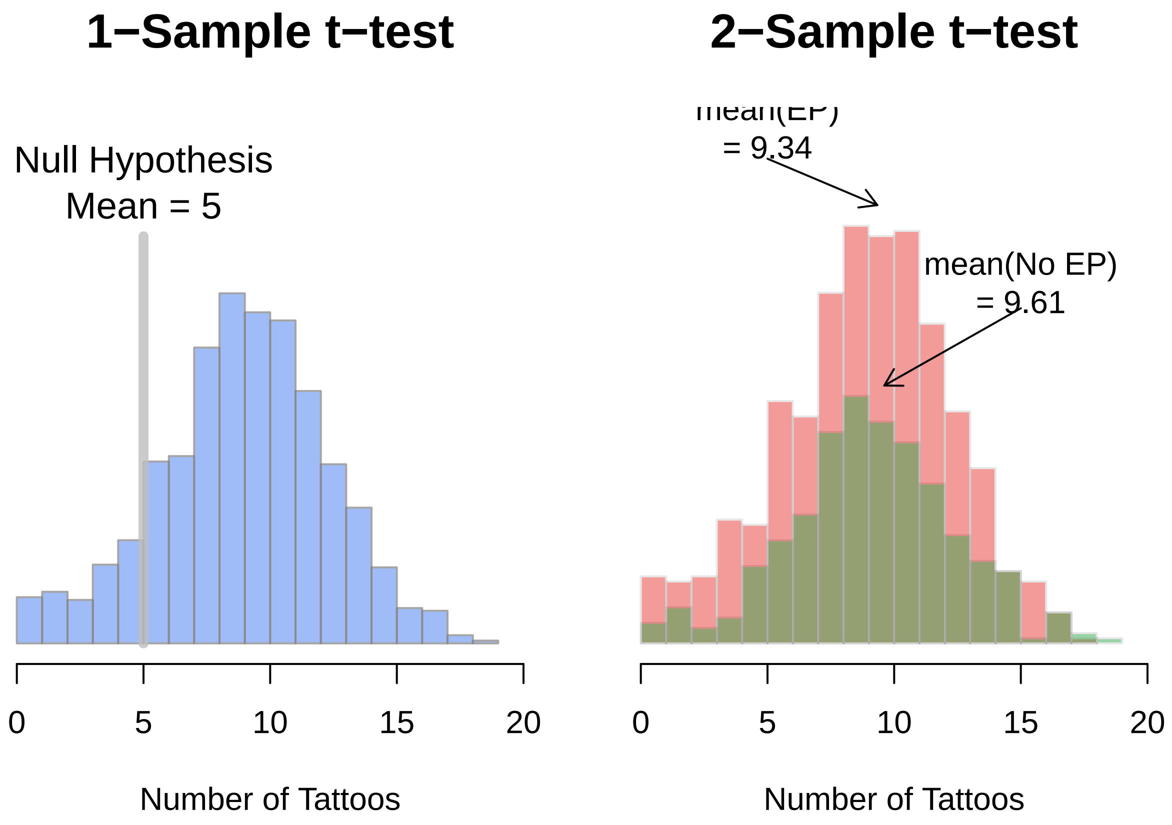
Now, if we want to access the test statistic or p-value directly, we can just use `$`:

```
# Get the test statistic
height.htest$statistic
##   t
## -21

# Get the p-value
height.htest$p.value
## [1] 1.4e-78

# Get a confidence interval for the mean
height.htest$conf.int
## [1] -15 -13
## attr(,"conf.level")
## [1] 0.95
```

13.3 T-test: `t.test()`



To compare the mean of 1 group to a specific value, or to compare the means of 2 groups, you do a *t-test*. The t-test function in R is `t.test()`. The `t.test()` function can take several arguments, here I'll emphasize a few of them. To see them all, check the help menu for t.test (`?t.test`).

13.3.1 1-sample t-test

Argument	Description
`x`	A vector of data whose mean you want to compare to the null hypothesis `mu`
`mu`	The population mean under the null hypothesis. For example, `mu = 0` will test the null hypothesis that the true population mean is 0.
`alternative`	A string specifying the alternative hypothesis. Can be `"two.sided"` indicating a two-tailed test, or `"greater"` or “`less"` for a one-tailed test.

In a one-sample t-test, you compare the data from one group of data to some hypothesized mean. For example, if someone said that pirates on average have 5 tattoos, we could conduct a one-sample test comparing the data from a sample of pirates to a hypothesized mean of 5. To conduct a one-sample t-test in R using `t.test()`, enter a vector as the main argument `x`, and the null hypothesis as the argument `mu`

Here, I'll conduct a t-test to see if the average number of tattoos owned by pirates is different from 5:

```
tattoo.ttest <- t.test(x = pirates$tattoos,  # Vector of data
                       mu = 5)               # Null: Mean is 5

# Print the result
tattoo.ttest
##
##  One Sample t-test
##
## data:  pirates$tattoos
## t = 40, df = 1000, p-value <2e-16
## alternative hypothesis: true mean is not equal to 5
## 95 percent confidence interval:
##  9.2 9.6
## sample estimates:
## mean of x
##       9.4
```

As you can see, the function printed lots of information: the sample mean was 9.43, the test statistic (41.59), and the p-value was `2e-16` (which is virtually 0). Because 2e-16 is less than 0.05, we would reject the null hypothesis that the true mean is equal to 5.

Now, what happens if I change the null hypothesis to a mean of 9.4? Because the sample mean was 9.43, quite close to 9.4, the test statistic should decrease, and the p-value should increase:

```
tattoo.ttest <- t.test(x = pirates$tattoos,
                       mu = 9.5)  # Null: Mean is 9.4

tattoo.ttest
##
##  One Sample t-test
##
## data:  pirates$tattoos
## t = -0.7, df = 1000, p-value = 0.5
## alternative hypothesis: true mean is not equal to 9.5
## 95 percent confidence interval:
##  9.2 9.6
## sample estimates:
## mean of x
##       9.4
```

Just as we predicted! The test statistic decreased to just -0.67, and the p-value increased to 0.51. In other words, our sample mean of 9.43 is reasonably consistent with the hypothesis that the true population mean is 9.50.

13.3.2 2-sample t-test

In a two-sample t-test, you compare the means of two groups of data and test whether or not they are the same. We can specify two-sample t-tests in one of two ways. If the dependent and independent variables are in a dataframe, you can use the formula notation in the form `y ~ x`, and specify the dataset containing the data in `data`

```
# Fomulation of a two-sample t-test

# Method 1: Formula
t.test(formula = y ~ x,  # Formula
       data = df) # Dataframe containing the variables
```

Alternatively, if the data you want to compare are in individual vectors (not together in a dataframe), you can use the vector notation:

```
# Method 2: Vector
t.test(x = x,  # First vector
       y = y)  # Second vector
```

For example, let's test a prediction that pirates who wear eye patches have fewer tattoos on average than those who don't wear eye patches. Because the data are in the `pirates` dataframe, we can do this using the formula method:

```
# 2-sample t-test
#  IV = eyepatch (0 or 1)
#  DV = tattoos

tat.patch.htest <- t.test(formula = tattoos ~ eyepatch,
                          data = pirates)
```

This test gave us a test statistic of 1.22 and a p-value of 0.22. Because the p-value is greater than 0.05, we would fail to reject the null hypothesis.

```
# Show me all of the elements in the htest object
names(tat.patch.htest)
## [1] "statistic"   "parameter"   "p.value"     "conf.int"    "estimate"
## [6] "null.value"  "alternative" "method"      "data.name"
```

Now, we can, for example, access the confidence interval for the mean differences using `$`

```
# Confidence interval for mean differences
tat.patch.htest$conf.int
## [1] -0.16  0.71
## attr(,"conf.level")
## [1] 0.95
```

13.3.2.1 Using subset to select levels of an IV

If your independent variable has more than two values, the `t.test()` function will return an error because it doesn't know which two groups you want to compare. For example, let's say I want to compare the number of tattoos of pirates of different ages. Now, the age column has many different values, so if I don't tell `t.test()` which two values of `age` I want to compare, I will get an error like this:

```
# Will return an error because there are more than
#  2 levels of the age IV

t.test(formula = tattoos ~ age,
       data = pirates)
```

To fix this, I need to tell the `t.test()` function which two values of `age` I want to test. To do this, use the `subset` argument and indicate which values of the IV you want to test using the `%in%` operator. For example, to compare the number of tattoos between pirates of age 29 and 30, I would add the `subset = age %in% c(29, 30)` argument like this:

```
# Compare the tattoos of pirates aged 29 and 30:
tatage.htest <- t.test(formula = tattoos ~ age,
                       data = pirates,
                       subset = age %in% c(29, 30))  # Compare age of 29 to 30

tatage.htest
##
##  Welch Two Sample t-test
##
## data:  tattoos by age
## t = 0.3, df = 100, p-value = 0.8
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.1  1.4
## sample estimates:
## mean in group 29 mean in group 30
##             10.1              9.9
```

Looks like we got a p-value of 0.79 which is pretty high and suggests that we should fail to reject the null hypothesis.

You can select any subset of data in the `subset` argument to the `t.test()` function – not just the primary independent variable. For example, if I wanted to compare the number of tattoos between pirates who wear headbands or not, but only for female pirates, I would do the following

```
# Is there an effect of college on # of tattoos
#  only for female pirates?

t.test(formula = tattoos ~ college,
       data = pirates,
       subset = sex == "female")
##
##  Welch Two Sample t-test
##
## data:  tattoos by college
## t = 1, df = 500, p-value = 0.3
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.27  0.92
## sample estimates:
##  mean in group CCCC mean in group JSSFP
##                 9.6                 9.3
```

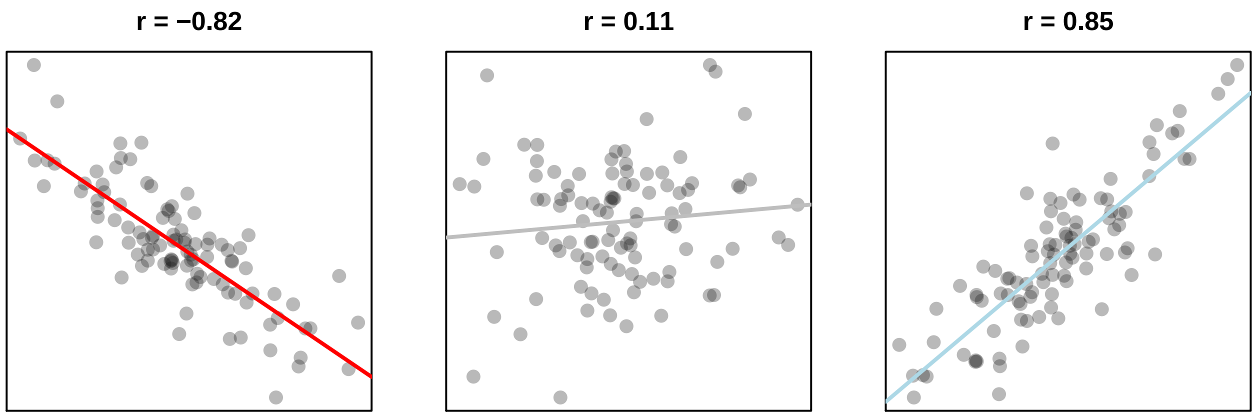


Figure 13.5: Three different correlations. A strong negative correlation, a very small positive correlation, and a strong positive correlation.

13.4 Correlation: `cor.test()`

Argument	Description
`formula`	A formula in the form `~ x + y`, where x and y are the names of the two variables you are testing. These variables should be two separate columns in a dataframe.
`data`	The dataframe containing the variables x and y
`alternative`	A string specifying the alternative hypothesis. Can be `"two.sided"` indicating a two-tailed test, or `"greater"` or “`less"` for a one-tailed test.
`method`	A string indicating which correlation coefficient to calculate and test. `"pearson"` (the default) stands for Pearson, while `"kendall"` and `"spearman"` stand for Kendall and Spearman correlations respectively.
`subset`	A vector specifying a subset of observations to use. E.g.; `subset = sex == "female"`

Next we'll cover two-sample correlation tests. In a correlation test, you are accessing the relationship between two variables on a ratio or interval scale: like height and weight, or income and beard length. The test statistic in a correlation test is called a *correlation coefficient* and is represented by the letter r. The coefficient can range from -1 to +1, with -1 meaning a strong negative relationship, and +1 meaning a strong positive relationship. The null hypothesis in a correlation test is a correlation of 0, which means no relationship at all:

To run a correlation test between two variables x and y, use the `cor.test()` function. You can do this in one of two ways, if x and y are columns in a dataframe, use the formula notation (`formula = ~ x + y`). If x and y are separate vectors (not in a dataframe), use the vector notation (`x, y`):

```
# Correlation Test
#   Correlating two variables x and y

# Method 1: Formula notation
##  Use if x and y are in a dataframe
cor.test(formula = ~ x + y,
         data = df)
```

```r
# Method 2: Vector notation
## Use if x and y are separate vectors
cor.test(x = x,
         y = y)
```

Let's conduct a correlation test on the `pirates` dataset to see if there is a relationship between a pirate's age and number of parrots they've had in their lifetime. Because the variables (age and parrots) are in a dataframe, we can do this in formula notation:

```r
# Is there a correlation between a pirate's age and
#  the number of parrots (s)he's owned?

# Method 1: Formula notation
age.parrots.ctest <- cor.test(formula = ~ age + parrots,
                              data = pirates)
# Print result
age.parrots.ctest
##
##  Pearson's product-moment correlation
##
## data:  age and parrots
## t = 6, df = 1000, p-value = 1e-09
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.13 0.25
## sample estimates:
##  cor
## 0.19
```

We can also do the same thing using vector notation – the results will be exactly the same:

```r
# Method 2: Vector notation
age.parrots.ctest <- cor.test(x = pirates$age,
                              y = pirates$parrots)

# Print result
age.parrots.ctest
##
##  Pearson's product-moment correlation
##
## data:  pirates$age and pirates$parrots
## t = 6, df = 1000, p-value = 1e-09
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.13 0.25
## sample estimates:
##  cor
## 0.19
```

Looks like we have a positive correlation of 0.19 and a very small p-value. To see what information we can extract for this test, let's run the command `names()` on the test object:

```r
names(age.parrots.ctest)
## [1] "statistic"   "parameter"   "p.value"     "estimate"    "null.value"
## [6] "alternative" "method"      "data.name"   "conf.int"
```

Looks like we've got a lot of information in this test object. As an example, let's look at the confidence interval for the population correlation coefficient:

```
#  95% confidence interval of the correlation
#   coefficient
age.parrots.ctest$conf.int
## [1] 0.13 0.25
## attr(,"conf.level")
## [1] 0.95
```

Just like the `t.test()` function, we can use the `subset` argument in the `cor.test()` function to conduct a test on a subset of the entire dataframe. For example, to run the same correlation test between a pirate's age and the number of parrot's she's owned, but *only* for female pirates, I can add the `subset = sex == "female"` argument:

```
# Is there a correlation between age and
#  number parrots ONLY for female pirates?

cor.test(formula = ~ age + parrots,
         data = pirates,
         subset = sex == "female")
##
##  Pearson's product-moment correlation
##
## data:  age and parrots
## t = 5, df = 500, p-value = 4e-06
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.12 0.30
## sample estimates:
##  cor
## 0.21
```

The results look pretty much identical. In other words, the strength of the relationship between a pirate's age and the number of parrot's they've owned is pretty much the same for female pirates and pirates in general.

13.5 Chi-square: `chsq.test()`

Next, we'll cover chi-square tests. In a chi-square test test, we test whether or not there is a difference in the rates of outcomes on a nominal scale (like sex, eye color, first name etc.). The test statistic of a chi-square text is χ^2 and can range from 0 to Infinity. The null-hypothesis of a chi-square test is that $\chi^2 = 0$ which means no relationship.

A key difference between the `chisq.test()` and the other hypothesis tests we've covered is that `chisq.test()` requires a *table* created using the `table()` function as its main argument. You'll see how this works when we get to the examples.

13.5.0.1 1-sample Chi-square test

If you conduct a 1-sample chi-square test, you are testing if there is a difference in the number of members of each category in the vector. Or in other words, are all category memberships equally prevalent? Here's the general form of a one-sample chi-square test:

```r
# General form of a one-sample chi-square test
chisq.test(x = table(x))
```

As you can see, the main argument to `chisq.test()` should be a *table* of values created using the `table()` function. For example, let's conduct a chi-square test to see if all pirate colleges are equally prevalent in the `pirates` data. We'll start by creating a table of the college data:

```r
# Frequency table of pirate colleges
table(pirates$college)
##
##  CCCC JSSFP
##   658   342
```

Just by looking at the table, it looks like pirates are much more likely to come from Captain Chunk's Cannon Crew (CCCC) than Jack Sparrow's School of Fashion and Piratery (JSSFP). For this reason, we should expect a very large test statistic and a very small p-value. Let's test it using the `chisq.test()` function.

```r
# Are all colleges equally prevelant?
college.cstest <- chisq.test(x = table(pirates$college))

college.cstest
##
##  Chi-squared test for given probabilities
##
## data:  table(pirates$college)
## X-squared = 100, df = 1, p-value <2e-16
```

Indeed, with a test statistic of 99.86 and a tiny p-value, we can safely reject the null hypothesis and conclude that certain college *are* more popular than others.

13.5.0.2 2-sample Chi-square test

If you want to see if the frequency of one nominal variable depends on a second nominal variable, you'd conduct a 2-sample chi-square test. For example, we might want to know if there is a relationship between the college a pirate went to, and whether or not he/she wears an eyepatch. We can get a contingency table of the data from the `pirates` dataframe as follows:

```r
# Do pirates that wear eyepatches have come from different colleges
#  than those that do not wear eyepatches?

table(pirates$eyepatch,
      pirates$college)
##
##     CCCC JSSFP
##   0  225   117
##   1  433   225
```

To conduct a chi-square test on these data, we will enter table of the two data vectors:

```r
# Is there a relationship between a pirate's
# college and whether or not they wear an eyepatch?

colpatch.cstest <- chisq.test(x = table(pirates$college,
                                        pirates$eyepatch))
```

```
colpatch.cstest
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  table(pirates$college, pirates$eyepatch)
## X-squared = 0, df = 1, p-value = 1
```

It looks like we got a test statistic of $\chi^2 = 0$ and a p-value of 1. At the traditional p = .05 threshold for significance, we would conclude that we fail to reject the null hypothesis and state that we do not have enough information to determine if pirates from different colleges differ in how likely they are to wear eye patches.

13.5.1 Getting APA-style conclusions with the apa function

Most people think that R pirates are a completely unhinged, drunken bunch of pillaging buffoons. But nothing could be further from the truth! R pirates are a very organized and formal people who like their statistical output to follow strict rules. The most famous rules are those written by the American Pirate Association (APA). These rules specify exactly how an R pirate should report the results of the most common hypothesis tests to her fellow pirates.

For example, in reporting a t-test, APA style dictates that the result should be in the form t(df) = X, p = Y (Z-tailed), where df is the degrees of freedom of the text, X is the test statistic, Y is the p-value, and Z is the number of tails in the test. Now you can of course read these values directly from the test result, but if you want to save some time and get the APA style conclusion quickly, just use the apa function. Here's how it works:

Consider the following two-sample t-test on the pirates dataset that compares whether or not there is a significant age difference between pirates who wear headbands and those who do not:

```
test.result <- t.test(age ~ headband,
                      data = pirates)
test.result
##
##  Welch Two Sample t-test
##
## data:  age by headband
## t = 0.4, df = 100, p-value = 0.7
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.0  1.5
## sample estimates:
##  mean in group no mean in group yes
##                28                27
```

It looks like the test statistic is 0.35, degrees of freedom is 135.47, and the p-value is 0.73. Let's see how the apa function gets these values directly from the test object:

```
yarrr::apa(test.result)
## [1] "mean difference = -0.22, t(135.47) = 0.35, p = 0.73 (2-tailed)"
```

As you can see, the apa function got the values we wanted and reported them in proper APA style. The apa function will even automatically adapt the output for Chi-Square and correlation tests if you enter such a test object. Let's see how it works on a correlation test where we correlate a pirate's age with the number of parrots she has owned:

```r
# Print an APA style conclusion of the correlation
#  between a pirate's age and # of parrots
age.parrots.ctest <- cor.test(formula = ~ age + parrots,
                              data = pirates)

# Pring result
age.parrots.ctest
##
##  Pearson's product-moment correlation
##
## data:  age and parrots
## t = 6, df = 1000, p-value = 1e-09
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.13 0.25
## sample estimates:
##  cor
## 0.19

# Print the apa style conclusion!
yarrr::apa(age.parrots.ctest)
## [1] "r = 0.19, t(998) = 6.13, p < 0.01 (2-tailed)"
```

The apa function has a few optional arguments that control things like the number of significant digits in the output, and the number of tails in the test. Run `?apa` to see all the options.

13.6 Test your R might!

The following questions are based on data from either the `movies` or the `pirates` dataset in the yarrr package. Make sure to load the package first to get access to the data!

1. Do male pirates have significantly longer beards than female pirates? Test this by conducting a t-test on the relevant data in the pirates dataset. (Hint: You'll have to select just the female and male pirates and remove the 'other' ones using `subset()`)
2. Are pirates whose favorite pixar movie is Up more or less likely to wear an eye patch than those whose favorite pixar movie is Inside Out? Test this by conducting a chi-square test on the relevant data in the pirates dataset. (Hint: Create a new dataframe that only contains data from pirates whose favorite move is either Up or Inside Out using `subset()`. Then do the test on this new dataframe.)
3. Do longer movies have significantly higher budgets than shorter movies? Answer this question by conducting a correlation test on the appropriate data in the movies dataset.
4. Do R rated movies earn significantly more money than PG-13 movies? Test this by conducting a t-test on the relevant data in the movies dataset.
5. Are certain movie genres significantly more common than others in the movies dataset? Test this by conducting a 1-sample chi-square test on the relevant data in the movies dataset.
6. Do sequels and non-sequels differ in their ratings? Test this by conducting a 2-sample chi-square test on the relevant data in the movies dataset.