Chapter 6. Correlation

Daehyeog Lee

2023-09-30

Table of contents

6.0. Importing libraries

```
library(ggplot2); library(boot); library(ggm); library(Hmisc); library(polycor)
```

```
Attaching package: 'Hmisc'

The following object is masked from 'package:ggm':

    rcorr

The following objects are masked from 'package:base':

    format.pval, units
```

6.1. What will this chapter tell me?

We can express the relationships between variables statistically by looking at two measures: *Covariance* and the *Correlation coefficient*.

6.2. Looking at relationships

The starting point with a correlation analysis is looking at some scatterplots of the variables we have measured.

6.3. How do we measure relationships?

6.3.1. A detour into the murky world of covariance

Variance of a single variable represents the average amount that the data vary from the mean.

If we are interested in whether two variables are related, then we are interested in whether changes in one variable are met with similar changes in the other variable. If there were a relationship between two variables, then as one variable deviates from its mean, the other variable should deviate from its mean in the same or the directly opposite way. By multiplying the deviations of one variable by the corresponding deviations of a second variable, we get what is known as the **cross-product deviations**. To get the average value of the combined deviations for the two variables, we can divide it by the number of observations. This is called **covariance**.

Calculating the covariance is a good way to access whether two variables are related to each other. However, one problem with covariance is that it depends upon the scales of measurement used.

6.3.2. Standardization and the correlation coefficient

Standardization makes us overcome the problem of dependence on the measurement scale. We need a unit of measurement into which any scale of measurement can be converted, and this unit of measurement we use is the *standard deviation*. By dividing any distance from the mean by SD, we obtain the distance in standard deviation units.

The standardized covariance is known as a **correlation coefficient (Pearson product-moment correlation coefficient).** A coefficient of +1 (-1) indicates that the two variables are perfectly positively (negatively) correlated. A coefficient of zero indicates no linear relationship at all and so if one variable changes, the other stays the same. Correlation coefficient values of +-.1 represent a small effect, +-.3 is a medium effect and +-.5 is a large effect.

6.3.3. The significance of the correlation coefficient

We may test the hypothesis that the correlation is different from zero. There are two ways to test this hypothesis.

The first is using a z-score. A problem here is that Pearson's r is known to have a sampling distribution that is not normally distributed. To deal with it, we can adjust r so that its sampling distribution is normal. To see whether the correlation is different from 0, we can subtract 0 from the observed value of r and divide by the standard error.

6.3.4. Confidence intervals for *r*

We can construct a confidence interval of transformed correlation coefficients just the way as computing confidence interval of the mean (X).

6.3.5. A world of warning about interpretation: causality

Correlation coefficient gives no indication of the direction of **causality**. In any correlation, Causality between two variables cannot be assumed because there may be other measured or unmeasured variables affecting the results. This is known as the **third-variableproblem** or the **tertium quid**. Even if we could ignore the third-variable problem, and even if we could assume that the two correlated variables were the only important ones, the correlation coefficient doesn't indicate in which direction causality operates.

6.4. Data entry for correlation analysis

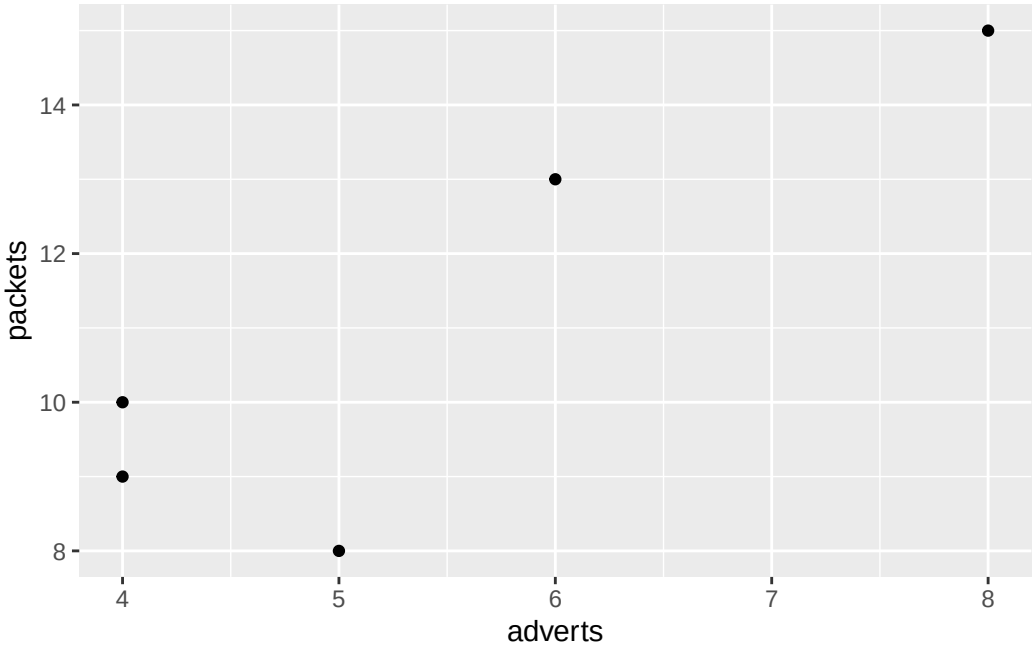
If we have a small dataset, we might want to enter the variables directly into **R** and then create a dataframe from them. From the advert data:

```
adverts <- c(5, 4, 4, 6, 8)
packets <- c(8, 9, 10, 13, 15)
advertData <- data.frame(adverts, packets)
advertData
```

```
  adverts packets
1       5       8
2       4       9
3       4      10
4       6      13
5       8      15
```

SELF-TEST

```
advertscatterplot <- ggplot(
  advertData,
  aes(adverts, packets
      )
) +
  geom_point(
)
advertscatterplot
```



6.5. Bivariate correlation

A **bivariate correlation** is a correlation between two variables whereas a **partial correlation** looks at the relationship between two variables while 'controlling' the effect of one or more additional variables. Pearson's product-moment correlation coefficient, Spearman's rho and Kendall's tau are examples of bivariate correlation coefficients.

6.5.1. Packages for correlation analysis in R

Packages *Hmisc, polycor, boot, ggplot2* and *ggm* are used in this chapter.

6.5.2. General procedure for correlations using R Commander

After importing the data in R Commander, we can use either the **Statistics -> Summaries -> Correlation matrix** or the **Statistics -> Summaries -> Correlation test** to get the correlation coefficients. The **correlation matrix** menu should be selected if we want to get correlation coefficients for more than two variables, and the **coefficient test** menu should be used when we want only a single correlation coefficient.

6.5.3. General procedure for correlations using R

`cor()`, `cor.test()` and `rcorr()` functions can be used to compute basic correlation coefficients. `cor.test()` enables us to get the confidence interval, but we can't compute multiple correlations with it. Below are the general forms of three functions

```
cor(x, y, use = "string", method = "correlation type"
```

use specifies how missing values are handled, and gets either "everything", "all.obs", or "complete.obs". *method* enables us to specify whether we want "pearson", "spearman", or "kendall" correlations.

```
rcorr(x, y, type = "correlation type")
```

type enables us to specify whether we want "pearson" or "spearman" correlations.

```
cor.test(x, y, alternative = "string", method = "correlation type", conf.level
= 0.95)
```

alternative specifies whether we want to do a two-tailed test (`alternavive = "two.sided"`), or whether we predict that the correlation will be less than zero or more than zero, in which case we can use `alternative = "less"` and `alternative = "greater"` respectively:

```
examData = read.delim("Exam Anxiety.dat", header = TRUE)
numeric_examData <- sapply(examData, is.numeric)
# cor() function
cor(examData[, numeric_examData], use = "complete.obs", method = "pearson")
```

```
               Code     Revise        Exam    Anxiety
Code     1.00000000 -0.2218286 -0.09779376  0.1135652
Revise  -0.22182864  1.0000000  0.39672070 -0.7092493
Exam    -0.09779376  0.3967207  1.00000000 -0.4409934
Anxiety  0.11356524 -0.7092493 -0.44099341  1.0000000
```

```
cor(examData$Exam, examData$Anxiety, use = "complete.obs", method = "pearson")
```

```
[1] -0.4409934
```

```
cor(examData$Exam, examData$Anxiety, use = "complete.obs", method = "kendall")
```

```
[1] -0.2847919
```

```r
cor(examData$Exam, examData$Anxiety, use = "pairwise.complete.obs",
    method = "kendall")
```

```
[1] -0.2847919
```

```r
# rcorr() function
rcorr(examData$Exam, examData$Anxiety, type = "pearson")
```

```
      x     y
x  1.00 -0.44
y -0.44  1.00

n= 103


P
  x  y
x     0
y  0
```

```r
# cor.test() function
cor.test(examData$Exam, examData$Anxiety, method = "pearson")
```

```
    Pearson's product-moment correlation

data:  examData$Exam and examData$Anxiety
t = -4.938, df = 101, p-value = 3.128e-06
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.5846244 -0.2705591
sample estimates:
       cor
-0.4409934
```

```r
cor.test(examData$Exam, examData$Anxiety, alternative = "less",
         method = "pearson")
```

```
    Pearson's product-moment correlation

data:  examData$Exam and examData$Anxiety
t = -4.938, df = 101, p-value = 1.564e-06
alternative hypothesis: true correlation is less than 0
95 percent confidence interval:
 -1.0000000 -0.2995071
sample estimates:
       cor
-0.4409934
```

```
cor.test(examData$Exam, examData$Anxiety, alternative = "less",
         method = "pearson", conf.level = 0.99)
```

```
    Pearson's product-moment correlation

data:  examData$Exam and examData$Anxiety
t = -4.938, df = 101, p-value = 1.564e-06
alternative hypothesis: true correlation is less than 0
99 percent confidence interval:
 -1.0000000 -0.2362782
sample estimates:
       cor
-0.4409934
```

6.5.4. Pearson's correlation coefficient

6.5.4.1. Assumptions of Pearson's *r*

Pearson's r was described in full at the beginning of this chapter. The sampling distribution has to be normally distributed. If our data are non-normal or not measured at the interval level, then we should use a different kind of correlation coefficient or use bootstrapping.

6.5.4.2. Computing Pearson's *r* using R

To compute Pearson's r from **Exam Anxiety.dat** file, we have to make a new dataframe by selecting only the variables of interest:

```
examData2 <- examData[, c("Exam", "Anxiety", "Revise")]
cor(examData2)
```

```
              Exam    Anxiety     Revise
Exam     1.0000000 -0.4409934  0.3967207
Anxiety -0.4409934  1.0000000 -0.7092493
Revise   0.3967207 -0.7092493  1.0000000
```

The first line creates a dataframe (*examData2*) that contains all of the cases, but only the variables **EXAM**, **Anxiety** and **Revise**. Alternatively, we could specify the subset of variable in the `examData` dataframe as part of the `cor()` function:

```
cor(examData[, c("Exam", "Anxiety", "Revise")])
```

```
              Exam    Anxiety     Revise
Exam     1.0000000 -0.4409934  0.3967207
Anxiety -0.4409934  1.0000000 -0.7092493
Revise   0.3967207 -0.7092493  1.0000000
```

The output provides a matrix of the correlation coefficients for the three variables. Each variable is perfectly correlated with itself, so $r = 1$ along the diagonal of the table.

Correlation coefficients are effect sizes, so we can interpret these values without worrying about p-values. However, if we want to get p-values, we can use the `rcorr()` function instead. To use the `rcorr()` function, we have to convert our dataframe into a matrix using the `as.matrix()` command:

```
examMatrix <- as.matrix(examData[, c("Exam", "Anxiety", "Revise")])
```

A matrix called `examMatrix` was created that contains only the variables **Exam**, **Anxiety**, and **Revise**. To get the correlation:

```
rcorr(examMatrix)
```

```
         Exam Anxiety Revise
Exam     1.00   -0.44   0.40
Anxiety -0.44    1.00  -0.71
Revise   0.40   -0.71   1.00

n= 103


P
        Exam Anxiety Revise
```

```
Exam          0       0
Anxiety  0            0
Revise   0    0
```

```
# or
rcorr(as.matrix(examData[, c("Exam", "Anxiety", "Revise")]))
```

```
         Exam Anxiety Revise
Exam     1.00   -0.44   0.40
Anxiety -0.44    1.00  -0.71
Revise   0.40   -0.71   1.00

n= 103


P
        Exam Anxiety Revise
Exam          0       0
Anxiety  0            0
Revise   0    0
```

The result shows that all of the correlation coefficients are significant.

To look at confidence intervals for correlation coefficients, we have to do this one at a time. For example, we can compute the confidence interval using `cor.test()` by executing:

```
cor.test(examData$Anxiety, examData$Exam)
```

```
    Pearson's product-moment correlation

data:  examData$Anxiety and examData$Exam
t = -4.938, df = 101, p-value = 3.128e-06
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.5846244 -0.2705591
sample estimates:
       cor
-0.4409934
```

As a result, the 95% confidence ranged from -.585 to -.271, which does not cross zero.

SELF-TEST

```
cor.test(examData$Revise, examData$Exam)
```

```
    Pearson's product-moment correlation

data:  examData$Revise and examData$Exam
t = 4.3434, df = 101, p-value = 3.343e-05
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.2200938 0.5481602
sample estimates:
      cor
0.3967207
```

```
cor.test(examData$Revise, examData$Anxiety)
```

```
    Pearson's product-moment correlation

data:  examData$Revise and examData$Anxiety
t = -10.111, df = 101, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.7938168 -0.5977733
sample estimates:
       cor
-0.7092493
```

6.5.4.3. Using R^2 for interpretation

The correlation coefficient squared (**coefficient of determination,** R^2) is a measure of the amount of variability in one variable that is shared by the other. If two variables, exam anxiety and exam performance had a correlation of -.4410 then the value of R^2 will be .194, which is 19.4%. We can say that exam anxiety shares 19.4% of the variability in exam performance. It leaves 80.6% of the variability still to be accounted for by other variables. Although R^2 is a useful measure of the substantive importance of an effect, it cannot be used to infer causal relationships.

For our `examData2` dataframe, we can get R^2 by executing:

```
cor(examData2)^2
```

```
             Exam   Anxiety    Revise
Exam    1.0000000 0.1944752 0.1573873
Anxiety 0.1944752 1.0000000 0.5030345
Revise  0.1573873 0.5030345 1.0000000
```

```r
cor(examData2)^2 * 100
```

```
             Exam   Anxiety    Revise
Exam    100.00000  19.44752  15.73873
Anxiety  19.44752 100.00000  50.30345
Revise   15.73873  50.30345 100.00000
```

6.5.5. Spearman's correlation coefficient

Spearman's correlation coefficient is a non-parametric statistic and so can be used when the data have violated parametric assumptions such as non-normally distributed data.

Let's perform Spearman correlation starting from importing **The Biggest Liar.dat**:

```r
liarData = read.delim("The Biggest Liar.dat", header = TRUE)
```

To obtain the correlation coefficient for a pair of variables we can execute:

```r
cor(liarData$Position, liarData$Creativity, method = "spearman")
```

```
[1] -0.3732184
```

If we want a significance value for this correlation, we could either use `rcorr()` (we have to first convert the dataframe to a matrix), or simply use `cor.test`:

```r
liarMatrix <- as.matrix(liarData[, c("Position", "Creativity")])
rcorr(liarMatrix)
```

```
           Position Creativity
Position       1.00      -0.31
Creativity    -0.31       1.00

n= 68
```

```
P
           Position Creativity
Position           0.0111
Creativity 0.0111
```

```
cor.test(
  liarData$Position, liarData$Creativity, alternative = "less",
  method = "spearman"
  )
```

```
Warning in cor.test.default(liarData$Position, liarData$Creativity, alternative
= "less", : Cannot compute exact p-value with ties
```

```
    Spearman's rank correlation rho

data:  liarData$Position and liarData$Creativity
S = 71948, p-value = 0.0008602
alternative hypothesis: true rho is less than 0
sample estimates:
       rho
-0.3732184
```

The correlation coefficient between the two variables is fairly large, and the significance value of this coefficient is very small. Therefore, it can be concluded that there is a significant relationship between creativity scores and how well someone did in the World's Biggest Liar competition.

6.5.6. Kendall's tau (non-parametric)

Kendall's tau is another non-parametric correlation and it should be used rather than Spearman's coefficient when we have small data set with a large number of tied ranks:

```
cor(liarData$Position, liarData$Creativity, method = "kendall")
```

```
[1] -0.3002413
```

```
cor.test(
  liarData$Position, liarData$Creativity, alternative = "less",
  method = "kendall"
  )
```

```
    Kendall's rank correlation tau

data:  liarData$Position and liarData$Creativity
z = -3.2252, p-value = 0.0006294
alternative hypothesis: true tau is less than 0
sample estimates:
       tau
-0.3002413
```

The output is much the same as for Spearman's correlation. Despite the difference in the correlation coefficients we can still interpret this result as being a highly significant relationship (because the significance value of 0.001) However, Kendall's value is a more accurate gauge of what the correlation in the population would be.

SELF-TEST

```
adverts <- c(5, 4, 4, 6, 8)
packets <- c(8, 9, 10, 13, 15)
cor.test(adverts, packets)
```

```
    Pearson's product-moment correlation

data:  adverts and packets
t = 3.0732, df = 3, p-value = 0.05443
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.0479747  0.9914236
sample estimates:
      cor
0.8711651
```

6.5.7. Bootstrapping correlations

The `boot()` function takes the general form: `object <- boot(data, function, replications)`, in which `data` specifies the dataframe to be used, `function` is a func-

tion that we write to tell `boot()` what we want to bootstrap, and `replications` is a numbers specifying how many bootstrap samples we want to take. If we want to bootstrap Kendall tau with liar data, then our function will be:

```
bootTau <- function(liarData,i)cor(
  liarData$Position[i], liarData$Creativity[i], use = "complete.obs",
  method = "kendall"
  )
```

The object called `bootTau` has created, and a variable `i` refers to a particular bootstrap sample. `cor()` was specified in exactly the same way as when we did the original Kendall correlation except that for each variable we have added `[i]`.

To create the bootstrap object, we execute:

```
boot_kendall <- boot(liarData, bootTau, 2000)
boot_kendall
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = liarData, statistic = bootTau, R = 2000)


Bootstrap Statistics :
      original       bias    std. error
t1* -0.3002413 -0.001658749   0.1000318
```

```
boot.ci(boot_kendall)
```

```
Warning in boot.ci(boot_kendall): bootstrap variances needed for studentized
intervals

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 2000 bootstrap replicates

CALL :
boot.ci(boot.out = boot_kendall)

Intervals :
```

```
Level      Normal              Basic
95%   (-0.4946, -0.1025 )   (-0.5012, -0.1098 )

Level     Percentile            BCa
95%   (-0.4907, -0.0993 )   (-0.4765, -0.0823 )
Calculations and Intervals on Original Scale
```

None of the confidence intervals cross zero, which gives us good reason to think that the population value of this relationship between creativity and success at being a liar is in the same direction as the sample value. In other words, our original conclusion stand.

SELF-TEST

```
bootPearson <- function(examData2,i)cor(
  examData2$Exam[i], examData2$Anxiety[i], use = "complete.obs",
  method = "pearson"
  )
boot_Pearson <- boot(examData2, bootPearson, 2000)
boot_Pearson
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = examData2, statistic = bootPearson, R = 2000)


Bootstrap Statistics :
      original      bias    std. error
t1* -0.4409934 0.001598284  0.06200801
```

```
bootSpearman <- function(examData2,i)cor(
  examData2$Exam[i], examData2$Anxiety[i], use = "complete.obs",
  method = "spearman"
  )
boot_Spearman <- boot(examData2, bootSpearman, 2000)
boot_Spearman
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP
```

```
Call:
boot(data = examData2, statistic = bootSpearman, R = 2000)


Bootstrap Statistics :
      original     bias    std. error
t1* -0.4046141 0.00297807  0.08036198
```

6.5.8. Biserial and point-biserial correlations

The biserial and point-biserial correlation coefficients are used when one of the two variables is **dichotomous**. The difference between the use of biserial and point-biserial correlations depends on whether the dichotomous variable is discrete or continuous. The **point-biserial correlation** coefficient is used when one variable is a discrete dichotomy (e.g., pregnent / not pregnent). The **biserial correlation** coefficient is used when one variable is a continuous dichotomy (e.g., passing or failing an exam; some people will just fail while others will fail by a large margin).

We can get a relationship between the gender of a cat and how much time it spent away from home using the **pbcorr.csv** file. It has three variables which are **time**, **gender**,and **recode**. We can carry out a Pearson correlation on **time** and **gender** by executing:

```
catData = read.csv("pbcorr.csv", header = TRUE)
cor.test(catData$time, catData$gender)
```

```
    Pearson's product-moment correlation

data:  catData$time and catData$gender
t = 3.1138, df = 58, p-value = 0.002868
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.137769 0.576936
sample estimates:
      cor
0.3784542
```

The output says the point-biserial correlation coefficient is 0.378, which has a significance value of 0.003. To prove that this is the case, we can carry out a Pearson correlation on **time** and **recode**:

```
cor.test(catData$time, catData$recode)
```

```
    Pearson's product-moment correlation

data:  catData$time and catData$recode
t = -3.1138, df = 58, p-value = 0.002868
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.576936 -0.137769
sample estimates:
       cor
-0.3784542
```

We can find that the sign has switched. Therefore, it can be inferred that the sign of the coefficient is completely dependent on which category we assign to which code.

If we want to convert the point-biserial correlation into the biserial correlation coefficient, we have to use the `table()` function to compute the frequencies of male and female cats. We then store the frequencies in a new object, and use it to compute the proportion of male and female cats using the `prop.table()` function:

```
catFrequencies <- table(catData$gender)
prop.table(catFrequencies)
```

```
        0         1
0.5333333 0.4666667
```

```
polyserial(catData$time, catData$gender)
```

```
[1] 0.4749256
```

The result from the `polyserial()` function indicates the biserial correlation coefficient.

6.6. Partial correlation

6.6.1. The theory behind part and partial correlation

Partial correlation is a correlation between two variables in which the effects of other variables are held constant.

6.6.2. Partial correlation using R

We will conduct a partial correlation between exam anxiety and exam performance while 'controlling' for the effect of revision time. To compute a partial correlation and its significance we will use the `pcor()` and `pcor.test()` functions respectively. The `pcor()` function takes the first two variables for the correlation, and any other variables are handled ad 'control':

```
pc <- pcor(c("Exam", "Anxiety", "Revise"), var(examData2))
pc
```

```
[1] -0.2466658
```

```
pc^2
```

```
[1] 0.06084403
```

The general form of `pcor.test()` is `pcor(pcor object, number of control variables, sample size)`. We created a partial correlation object called `pc`, had only one control variable (`Revise`) and there was a sample size of 103. Therefore we can execute:

```
pcor.test(pc, 1, 103)
```

```
$tval
[1] -2.545307

$df
[1] 100

$pvalue
[1] 0.01244581
```

The output of `pcor()` is the partial correlation for the variables `Anxiety` and `Exam` but controlling for the effect of `Revision`. The partial correlation between exam performance and exam anxiety is -.247, which is considerably less than the correlation when the effect of revision time is not controlled for (r=-.441). Although this correlation is still statistically significant, the relationship is diminished.

6.6.3. Semi-partial (or part) correlations

In a **semi-partial** correlation we control for the effect that the third variable has on only one of the variables in the correlation.

6.7. Comparing correlations

6.7.1. Comparing independent *r*s

Sometimes we want to know whether one correlation coefficient is bigger than another. We could compute the correlation in these two samples by using the `subset()` function:

```
mExam <- subset(examData, Gender == "Male", select = c("Exam", "Anxiety"))
fExam <- subset(examData, Gender == "Female", select = c("Exam", "Anxiety"))
cor(mExam)
```

```
              Exam    Anxiety
Exam     1.0000000 -0.5056874
Anxiety -0.5056874  1.0000000
```

```
cor(fExam)
```

```
              Exam    Anxiety
Exam     1.0000000 -0.3813845
Anxiety -0.3813845  1.0000000
```

We can compare the correlation coefficient by converting these to z scores.

6.7.2. Comparing dependent *r*s

If we want to compare correlation coefficients that come from the same entities, we have to use a t-statistic to test whether a difference between two dependent correlations from the same sample is significant.

6.8. Calculating the effect size

Although the Spearman and Kendall correlations are comparable in many respects, there are two important differences.

First, a squared Spearman's r is usually a good approximation for squared Pearson's r. However, squared Kendall's *tau* does not tell us about the proportion of variance shared by two variables.

Second, Spearman's r and Pearson's r are generally similar sizes, but Kendall's *tau* is 66~75% smaller.

6.9. How to report correlation coefficients

Examples

There was a significant relationship between the number of adverts watched and the number of packets of sweets purchased, $r = .87$, p (one-tailed) < 0.5.