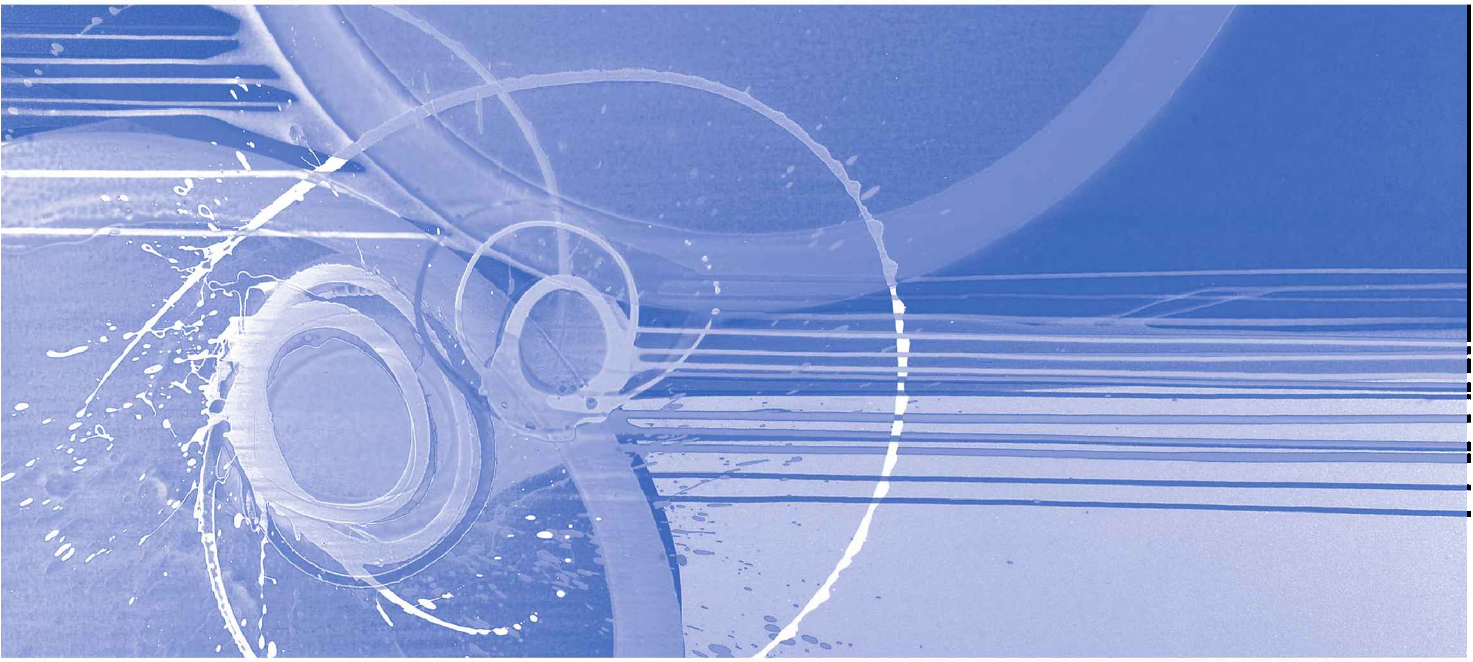
DISCOVERING STATISTICS USING R



ANDY FIELD | JEREMY MILES | ZOË FIELD

SAGE

Los Angeles | London | New Delhi
Singapore | Washington DC

First published 2012

SAGE Publications Ltd
1 Oliver's Yard
55 City Road
London EC1Y 1SP

SAGE Publications Inc.
2455 Teller Road
Thousand Oaks, California 91320

SAGE Publications India Pvt Ltd
B 1/I 1 Mohan Cooperative Industrial Area
Mathura Road
New Delhi 110 044

SAGE Publications Asia-Pacific Pte Ltd
3 Church Street
#10-04 Samsung Hub
Singapore 049483

Library of Congress Control Number: Available

British Library Cataloguing in Publication data

A catalogue record for this book is available from the British Library

ISBN 978-1-4462-0045-2
ISBN 978-1-4462-0046-9

Typeset by C&M Digitals (P) Ltd, Chennai, India
Printed and bound in Great Britain by Ashford Colour Press Ltd
Printed on paper from sustainable resources

Correlation

6

FIGURE 6.1
I don't have a photo from Christmas 1981, but this was taken about that time at my grandparents' house. I'm trying to play an 'E' by the looks of it, no doubt because it's in 'Take on the World'.

6.1. What will this chapter tell me? ①

When I was 8 years old, my parents bought me a guitar for Christmas. Even then, I'd desperately wanted to play the guitar for years. I could not contain my excitement at getting this gift (had it been an *electric* guitar I think I would have actually exploded with excitement). The guitar came with a 'learn to play' book and, after a little while of trying to play what was on page 1 of this book, I readied myself to unleash a riff of universe-crushing power onto the world (well, 'Skip to my Lou' actually). But, I couldn't do it. I burst into tears and ran upstairs to hide.[1] My dad sat with me and said 'Don't worry, Andy, everything is hard to begin with, but the more you practise the easier it gets.' In his comforting words, my dad was inadvertently teaching me about the relationship, or correlation, between two variables. These two variables could be related in three ways: (1) *positively related*, meaning that the more I practised my guitar, the better a guitar player I would become (i.e., my dad was telling me the truth); (2) *not related* at all, meaning that as I practised the guitar my playing ability would remain completely constant (i.e., my dad has fathered a cretin); or (3) *negatively related*, which would mean that the more I practised my guitar the worse a guitar player I would become (i.e., my dad has fathered an indescribably strange child). This chapter looks first at how we can express the relationships between variables statistically by looking at two measures: *covariance* and the *correlation coefficient*. We then discover how to carry out and interpret correlations in **R**. The chapter ends by looking at more complex measures of relationships; in doing so it acts as a precursor to multiple regression, which we discuss in Chapter 7.

6.2. Looking at relationships ①

In Chapter 4 I stressed the importance of looking at your data graphically before running any other analysis on them. I just want to begin by reminding you that our first starting point with a correlation analysis should be to look at some scatterplots of the variables we have measured. I am not going to repeat how to get **R** to produce these graphs, but I am going to urge you (if you haven't done so already) to read section 4.5 before embarking on the rest of this chapter.

6.3. How do we measure relationships? ①

6.3.1. A detour into the murky world of covariance ①

The simplest way to look at whether two variables are associated is to look at whether they *covary*. To understand what **covariance** is, we first need to think back to the concept of variance that we met in Chapter 2. Remember that the variance of a single variable represents the average amount that the data vary from the mean. Numerically, it is described by:

$$\text{Variance}(s^2) = \frac{\sum(x_i - \bar{x})^2}{N-1} = \frac{\sum(x_i - \bar{x})(x_i - \bar{x})}{N-1} \tag{6.1}$$

The mean of the sample is represented by $\bar{x}$, x_i is the data point in question and N is the number of observations (see section 2.4.1). 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. Therefore, when one variable deviates from its mean we would expect the other variable to deviate from its mean in a similar way. To illustrate what I mean, imagine we took five people and subjected them to a certain number of advertisements promoting toffee sweets, and then measured how many packets of those sweets each

[1] This is not a dissimilar reaction to the one I have when publishers ask me for new editions of statistics textbooks.

Table 6.1 Adverts watched and toffee purchases

Participant:	*1*	*2*	*3*	*4*	*5*	*Mean*	*s*
Adverts watched	5	4	4	6	8	5.4	1.67
Packets bought	8	9	10	13	15	11.0	2.92

person bought during the next week. The data are in Table 6.1 as well as the mean and standard deviation (*s*) of each variable.

If there were a relationship between these 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. Figure 6.2 shows the data for each participant (light blue circles represent the number of packets bought and dark blue circles represent the number of adverts watched); the grey line is the average number of packets bought and the blue line is the average number of adverts watched. The vertical lines represent the differences (remember that these differences are called *deviations*) between the observed values and the mean of the relevant variable. The first thing to notice about Figure 6.2 is that there is a very similar pattern of deviations for both variables. For the first three participants the observed values are below the mean for both variables, for the last two people the observed values are above the mean for both variables. This pattern is indicative of a potential relationship between the two variables (because it seems that if a person's score is below the mean for one variable then their score for the other will also be below the mean).

So, how do we calculate the exact similarity between the patterns of differences of the two variables displayed in Figure 6.2? One possibility is to calculate the total amount of deviation but we would have the same problem as in the single variable case: the positive and negative deviations would cancel out (see section 2.4.1). Also, by simply adding the deviations, we would gain little insight into the relationship between the variables. Now, in the single variable case, we squared the deviations to eliminate the problem of positive and negative deviations cancelling out each other. When there are two variables, rather than squaring each deviation, we can multiply the deviation for one variable by the corresponding deviation for the second variable. If both deviations are positive or negative then this will give us a positive value (indicative of the deviations being in the same direction), but

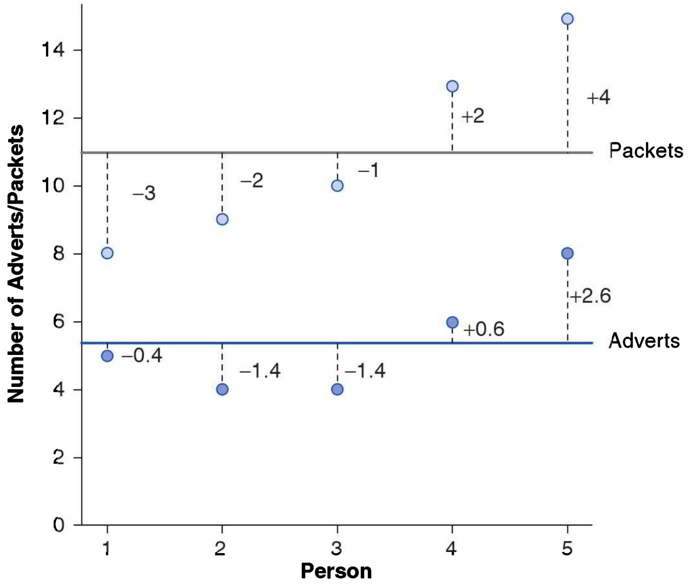


FIGURE 6.2 Graphical display of the differences between the observed data and the means of two variables

if one deviation is positive and one negative then the resulting product will be negative (indicative of the deviations being opposite in direction). When we multiply the deviations of one variable by the corresponding deviations of a second variable, we get what is known as the **cross-product deviations.** As with the variance, if we want an average value of the combined deviations for the two variables, we must divide by the number of observations (we actually divide by $N - 1$ for reasons explained in Jane Superbrain Box 2.2). This averaged sum of combined deviations is known as the **covariance.** We can write the covariance in equation form as in equation (6.2) – you will notice that the equation is the same as the equation for variance, except that instead of squaring the differences, we multiply them by the corresponding difference of the second variable:

$$\mathrm{cov}(x,y) = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{N-1} \tag{6.2}$$

For the data in Table 6.1 and Figure 6.2 we reach the following value:

$$\begin{aligned}\mathrm{cov}(x,y) &= \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{N-1}\\ &= \frac{(-0.4)(-3)+(-1.4)(-2)+(-1.4)(-1)+(0.6)(2)+(2.6)(4)}{4}\\ &= \frac{1.2+2.8+1.4+1.2+10.4}{4}\\ &= \frac{17}{4}\\ &= 4.25\end{aligned}$$

Calculating the covariance is a good way to assess whether two variables are related to each other. A positive covariance indicates that as one variable deviates from the mean, the other variable deviates in the same direction. On the other hand, a negative covariance indicates that as one variable deviates from the mean (e.g., increases), the other deviates from the mean in the opposite direction (e.g., decreases).

There is, however, one problem with covariance as a measure of the relationship between variables and that is that it depends upon the scales of measurement used. So, covariance is not a standardized measure. For example, if we use the data above and assume that they represented two variables measured in miles then the covariance is 4.25 (as calculated above). If we then convert these data into kilometres (by multiplying all values by 1.609) and calculate the covariance again then we should find that it increases to 11. This dependence on the scale of measurement is a problem because it means that we cannot compare covariances in an objective way – so, we cannot say whether a covariance is particularly large or small relative to another data set unless both data sets were measured in the same units.

6.3.2. Standardization and the correlation coefficient ①

To overcome the problem of dependence on the measurement scale, we need to convert the covariance into a standard set of units. This process is known as **standardization.** A very basic form of standardization would be to insist that all experiments use the same units of measurement, say metres – that way, all results could be easily compared. However, what happens if you want to measure attitudes – you'd be hard pushed to measure them in metres. Therefore, we need a unit of measurement into which any scale of measurement can be converted. The unit of measurement we use is the *standard deviation*. We came across this measure in section 2.4.1 and saw that, like the variance, it is a measure of the average deviation from the mean. If we divide any distance from the mean by the standard deviation, it gives us that distance in standard deviation units. For example, for the data in Table 6.1, the standard deviation for the number of packets bought is approximately 3.0 (the exact value is 2.92). In Figure 6.2 we can see that the observed value for participant 1 was 3 packets less than the mean (so there was an error of −3 packets of sweets). If we divide this deviation, −3, by the standard deviation, which is approximately 3, then we get a value of −1. This tells us that the difference between participant 1's score and the mean was −1 standard deviation. So, we can express the deviation from the mean for a participant in standard units by dividing the observed deviation by the standard deviation.

It follows from this logic that if we want to express the covariance in a standard unit of measurement we can simply divide by the standard deviation. However, there are two variables and, hence, two standard deviations. Now, when we calculate the covariance we actually calculate two deviations (one for each variable) and then multiply them. Therefore, we do the same for the standard deviations: we multiply them and divide by the product of this multiplication. The standardized covariance is known as a **correlation coefficient** and is defined by equation (6.3), in which s_x is the standard deviation of the first variable and s_y is the standard deviation of the second variable (all other letters are the same as in the equation defining covariance):

$$r = \frac{\mathrm{cov}_{xy}}{s_x s_y} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{(N-1)s_x s_y} \tag{6.3}$$

The coefficient in equation (6.3) is known as the **Pearson product-moment correlation coefficient** or **Pearson correlation coefficient** (for a really nice explanation of why it was originally called the 'product-moment' correlation, see Miles & Banyard, 2007) and was invented by Karl Pearson (see Jane Superbrain Box 6.1).[2] If we look back at Table 6.1 we see that the standard deviation for the number of adverts watched (s_x) was 1.67, and for the number of packets of crisps bought (s_y) was 2.92. If we multiply these together we get 1.67 × 2.92 = 4.88. Now, all we need to do is take the covariance, which we calculated a few pages ago as being 4.25, and divide by these multiplied standard deviations. This gives us r = 4.25/4.88 = .87.

By standardizing the covariance we end up with a value that has to lie between −1 and +1 (if you find a correlation coefficient less than −1 or more than +1 you can be sure that something has gone hideously wrong!). A coefficient of +1 indicates that the two variables are perfectly positively correlated, so as one variable increases, the other increases by a proportionate amount. Conversely, a coefficient of −1 indicates a perfect negative relationship: if one variable increases, the other decreases by a proportionate amount. A coefficient of zero indicates no linear relationship at all and so if one variable changes, the other stays the same. We also saw in section 2.6.4 that because the correlation coefficient is a standardized measure of an observed effect, it is a commonly used measure of the size of an effect and that values of ±.1 represent a small effect, ±.3 is a medium effect and ±.5 is a large effect (although I re-emphasize my caveat that these canned effect sizes are no substitute for interpreting the effect size within the context of the research literature).

[2] You will find Pearson's product-moment correlation coefficient denoted by both *r* and *R*. Typically, the uppercase form is used in the context of regression because it represents the multiple correlation coefficient; however, for some reason, when we square *r* (as in section 6.5.4.3) an upper case *R* is used. Don't ask me why – it's just to confuse me, I suspect.

JANE SUPERBRAIN 6.1

Who said statistics was dull? ①

Students often think that statistics is dull, but back in the early 1900s it was anything but dull, with various prominent figures entering into feuds on a soap opera scale. One of the most famous was between Karl Pearson and Ronald Fisher (whom we met in Chapter 2). It began when Pearson published a paper of Fisher's in his journal but made comments in his editorial that, to the casual reader, belittled Fisher's work. Two years later Pearson's group published work following on from Fisher's paper without consulting him. The antagonism persisted with Fisher turning down a job to work in Pearson's group and publishing 'improvements' on Pearson's ideas. Pearson for his part wrote in his own journal about apparent errors made by Fisher.

Another prominent statistician, Jerzy Neyman, criticized some of Fisher's most important work in a paper delivered to the Royal Statistical Society on 28 March 1935 at which Fisher was present. Fisher's discussion of the paper at that meeting directly attacked Neyman. Fisher more or less said that Neyman didn't know what he was talking about and didn't understand the background material on which his work was based. Relations soured so much that while they both worked at University College London, Neyman openly attacked many of Fisher's ideas in lectures to his students. The two feuding groups even took afternoon tea (a common practice in the British academic community of the time) in the same room but at different times! The truth behind who fuelled these feuds is, perhaps, lost in the mists of time, but Zabell (1992) makes a sterling effort to unearth it.

Basically, then, the founders of modern statistical methods were a bunch of squabbling children. Nevertheless, these three men were astonishingly gifted individuals. Fisher, in particular, was a world leader in genetics, biology and medicine as well as possibly the most original mathematical thinker ever (Barnard, 1963; Field, 2005c; Savage, 1976).

6.3.2. The significance of the correlation coefficient ③

Although we can directly interpret the size of a correlation coefficient, we have seen in Chapter 2 that scientists like to test hypotheses using probabilities. In the case of a correlation coefficient we can test the hypothesis that the correlation is different from zero (i.e., different from 'no relationship'). If we find that our observed coefficient was very unlikely to happen if there was no effect in the population, then we can gain confidence that the relationship that we have observed is statistically meaningful.

There are two ways that we can go about testing this hypothesis. The first is to use our trusty *z*-scores that keep cropping up in this book. As we have seen, *z*-scores are useful because we know the probability of a given value of *z* occurring, if the distribution from which it comes is normal. There is one problem with Pearson's *r*, which is that it is known to have a sampling distribution that is not normally distributed. This is a bit of a nuisance, but luckily, thanks to our friend Fisher, we can adjust *r* so that its sampling distribution *is* normal as follows (Fisher, 1921):

$$z_r = \frac{1}{2}\log_e\left(\frac{1+r}{1-r}\right) \tag{6.4}$$

The resulting z_r has a standard error of:

$$SE_{z_r} = \frac{1}{\sqrt{N-3}} \tag{6.5}$$

For our advert example, our $r = .87$ becomes 1.33 with a standard error of .71.

We can then transform this adjusted r into a z-score just as we have done for raw scores, and for skewness and kurtosis values in previous chapters. If we want a z-score that represents the size of the correlation relative to a particular value, then we simply compute a z-score using the value that we want to test against and the standard error. Normally we want to see whether the correlation is different from 0, in which case we can subtract 0 from the observed value of r and divide by the standard error (in other words, we just divide z_r by its standard error):

$$z = \frac{z_r}{SE_{z_r}} \tag{6.6}$$

For our advert data this gives us 1.33/.71 = 1.87. We can look up this value of z (1.87) in the table for the normal distribution in the Appendix and get the one-tailed probability from the column labelled 'Smaller Portion'. In this case the value is .0307. To get the two-tailed probability we simply multiply the one-tailed probability value by 2, which gives us .0614. As such the correlation is significant, $p < .05$, one-tailed, but not two-tailed.

In fact, the hypothesis that the correlation coefficient is different from 0 is usually (**R**, for example, does this) tested not using a z-score, but using a t-statistic with $N - 2$ degrees of freedom, which can be directly obtained from r:

$$t_r = \frac{r\sqrt{N-2}}{\sqrt{1-r^2}} \tag{6.7}$$

You might wonder then why I told you about z-scores, then. Partly it was to keep the discussion framed in concepts with which you are already familiar (we don't encounter the t-test properly for a few chapters), but also it is useful background information for the next section.

6.3.4. Confidence intervals for *r* ③

Confidence intervals tell us something about the likely value (in this case of the correlation) in the population. To understand how confidence intervals are computed for r, we need to take advantage of what we learnt in the previous section about converting r to z_r (to make the sampling distribution normal), and using the associated standard errors. We can then construct a confidence interval in the usual way. For a 95% confidence interval we have (see section 2.5.2.1):

$$\text{lower boundary of confidence interval} = \bar{X} - (1.96 \times SE)$$

$$\text{upper boundary of confidence interval} = \bar{X} + (1.96 \times SE)$$

In the case of our transformed correlation coefficients these equations become:

$$\text{lower boundary of confidence interval} = z_r - (1.96 \times SE_{z_r})$$

$$\text{upper boundary of confidence interval} = z_r + (1.96 \times SE_{z_r})$$

For our advert data this gives us 1.33 – (1.96 × .71) = –0.062, and 1.33 + (1.96 × .71) = 2.72. Remember that these values are in the z_r metric and so we have to convert back to correlation coefficients using:

$$r = \frac{e^{(2z_r)} - 1}{e^{(2z_r)} + 1} \tag{6.8}$$

This gives us an upper bound of r = .991 and a lower bound of –0.062 (because this value is so close to zero the transformation to z has no impact).

CRAMMING SAM'S TIPS **Correlation**

- A crude measure of the relationship between variables is the *covariance*.
- If we standardize this value we get *Pearson's correlation coefficient, r*.
- The correlation coefficient has to lie between –1 and +1.
- A coefficient of +1 indicates a perfect positive relationship, a coefficient of –1 indicates a perfect negative relationship, and a coefficient of 0 indicates no linear relationship at all.
- The correlation coefficient is a commonly used measure of the size of an effect: values of ±.1 represent a small effect, ±.3 is a medium effect and ±.5 is a large effect. However, if you can, try to interpret the size of correlation within the context of the research you've done rather than blindly following these benchmarks.

6.3.1. A word of warning about interpretation: causality ①

Considerable caution must be taken when interpreting correlation coefficients because they give no indication of the direction of *causality*. So, in our example, although we can conclude that as the number of adverts watched increases, the number of packets of toffees bought increases also, we cannot say that watching adverts *causes* you to buy packets of toffees. This caution is for two reasons:

- **The third-variable problem**: We came across this problem in section 1.6.2. To recap, 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-variable* problem or the *tertium quid* (see section 1.6.2 and Jane Superbrain Box 1.1).
- **Direction of causality**: Correlation coefficients say nothing about which variable causes the other to change. Even if we could ignore the third-variable problem described above, and we could assume that the two correlated variables were the only important ones, the correlation coefficient doesn't indicate in which direction causality operates. So, although it is intuitively appealing to conclude that watching adverts causes us to buy packets of toffees, there is no *statistical* reason why buying packets of toffees cannot cause us to watch more adverts. Although the latter conclusion makes less intuitive sense, the correlation coefficient does not tell us that it isn't true.

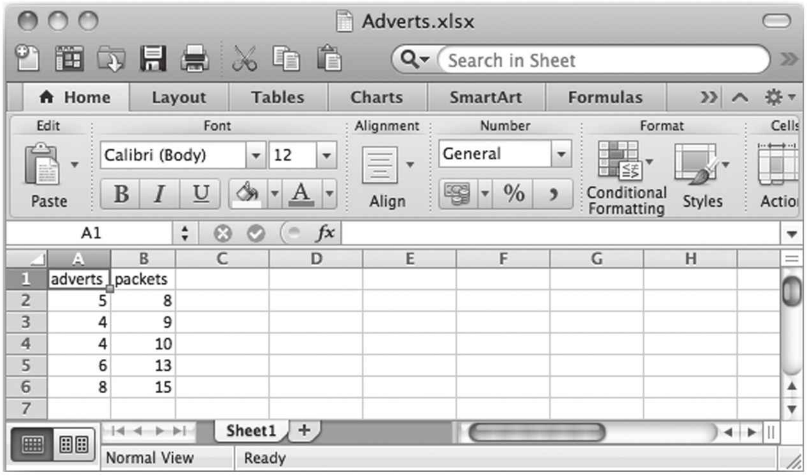
6.4. Data entry for correlation analysis ①

Data entry for correlation, regression and multiple regression is straightforward because each variable is entered in a separate column. If you are preparing your data in software other than **R** then this means that, for each variable you have measured, you create a variable in the spreadsheet with an appropriate name, and enter a participant's scores across one row of the spreadsheet. There may be occasions on which you have one or more categorical variables (such as gender) and these variables can also be entered in a column – see section 3.7 for more detail.

As an example, if we wanted to calculate the correlation between the two variables in Table 6.1 we would enter these data as in Figure 6.3. You can see that each variable is entered in a separate column, and each row represents a single individual's data (so the first consumer saw 5 adverts and bought 8 packets).

If you have a small data set you might want to enter the variables directly into **R** and then create a dataframe from them. For the advert data this can be done by executing the following commands (see section 3.5):

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



	A	B	C	D	E	F	G	H
1	adverts	packets						
2	5	8						
3	4	9						
4	4	10						
5	6	13						
6	8	15						
7								

FIGURE 6.3
Data entry for correlation using Excel

SELF-TEST

✓ Enter the advert data and use *ggplot2* to produce a scatterplot (number of packets bought on the *y*-axis, and adverts watched on the *x*-axis) of the data.

6.5. Bivariate correlation ①

There are two types of correlation: *bivariate* and *partial*. A **bivariate correlation** is a correlation between two variables (as described at the beginning of this chapter) whereas a **partial correlation** (see section 6.6) looks at the relationship between two variables while 'controlling' the effect of one or more additional variables. Pearson's product-moment correlation coefficient (described earlier), Spearman's rho (see section 6.5.5) and Kendall's tau (see section 6.5.6) are examples of bivariate correlation coefficients.

Let's return to the example from Chapter 4 about exam scores. Remember that a psychologist was interested in the effects of exam stress and revision on exam performance. She had devised and validated a questionnaire to assess state anxiety relating to exams (called the Exam Anxiety Questionnaire, or EAQ). This scale produced a measure of anxiety scored out of 100. Anxiety was measured before an exam, and the percentage mark of each student on the exam was used to assess the exam performance. She also measured the number of hours spent revising. These data are in **Exam Anxiety.dat** on the companion website. We already created scatterplots for these data (section 4.5) so we don't need to do that again.

6.3.1. Packages for correlation analysis in R ①

There are several packages that we will use in this chapter. Some of them can be accessed through R Commander (see the next section) but others can't. For the examples in this chapter you will need the packages *Hmisc*, *polycor*, *boot*, *ggplot2* and *ggm*. If you do not have these packages installed (some should be installed from previous chapters), you can install them by executing the following commands (*boot* is part of the base package and doesn't need to be installed):

```
install.packages("Hmisc"); install.packages("ggm");
install.packages("ggplot2"); install.packages("polycor")
```

You then need to load these packages by executing the commands:

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

6.3.2. General procedure for correlations using R Commander ①

To conduct a bivariate correlation using R Commander, first initiate the package by executing (and install it if you haven't – see section 3.6):

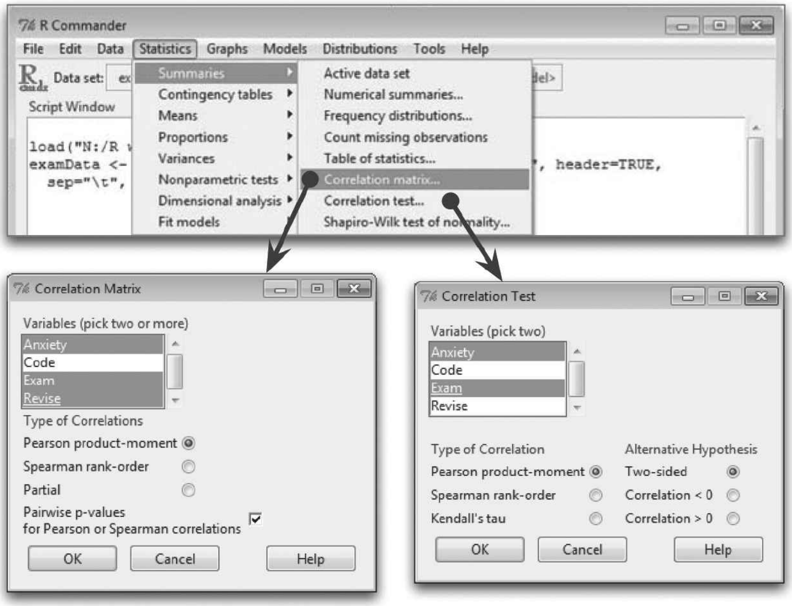
```
library(Rcmdr)
```

You then need to load the data file into R Commander by using the **Data⇒Import data⇒from text file, clipboard, or URL...** menu (see section 3.7.3). Once the data are loaded in a dataframe (I have called the dataframe *examData*), you can use either the **Statis tics⇒Summaries⇒Correlation matrix...** or the **Statistics⇒Summaries⇒Correlation test...** menu to get the correlation coefficients. These menus and their dialog boxes are shown in Figure 6.4.

The *correlation matrix* menu should be selected if you want to get correlation coefficients for more than two variables (in other words, produce a grid of correlation coefficients); the *correlation test* menu should be used when you want only a single correlation coefficient. Both menus enable you to compute Pearson's product-moment correlation and Spearman's correlation, and both can be used to produce *p*-values associated with these correlations. However, there are some important differences too: the correlation test menu enables you to compute Kendall's correlation, produces a confidence interval and allows you to select both two-tailed and one-tailed tests, but can be used to compute only one correlation coefficient at a time; in contrast, the correlation matrix cannot produce Kendall's correlation but can compute partial correlations, and can also compute multiple correlations from a single command.

Let's look at the *Correlation Matrix* dialog box first. Having accessed the main dialog box, you should find that the variables in the dataframe are listed on the left-hand side of the dialog box (Figure 6.4). You can select the variables that you want from the list by clicking with the mouse while holding down the *Ctrl* key. **R** will create a grid of correlation coefficients for all of the combinations of variables that you have selected. This table is called a correlation matrix. For our current example, select the variables **Exam**, **Anxiety** and **Revise**. Having selected the variables of interest you can choose between three correlation coefficients: Pearson's product-moment correlation coefficient (Pearson product-moment ◉), Spearman's rho (Spearman rank-order ◉) and a partial correlation (Partial ◉). Any of these can be selected by clicking on the appropriate tick-box with a mouse. Finally, if you would like *p*-values for the correlation coefficients then select[3] Pairwise p-values for Pearson or Spearman correlations ☑.

For the *correlation test* dialog box you will again find that the variables in the dataframe are listed on the left-hand side of the dialog box (Figure 6.4). You can select only two by clicking with the mouse while holding down the *Ctrl* key. Having selected the two variables of interest, choose between three correlation coefficients: Pearson's product-moment correlation coefficient (Pearson product-moment ◉), Spearman's rho (Spearman rank-order ◉) and Kendall's tau (Kendall's tau ◉). In addition, it is possible to specify whether or not the test is one- or two-tailed (see section 2.6.2). To recap, a two-tailed test (the default) should be used when you cannot predict the nature of the relationship (i.e., 'I'm not sure whether exam anxiety will improve or reduce exam marks'). If you have a non-directional hypothesis like this, click



```
load("N:/R
examData <-                                   , header=TRUE,
  sep="\t",
```

FIGURE 6.4 Conducting a bivariate correlation using R Commander

[3] Selecting this option changes the function that R Commander uses to generate the output. If this option is not selected then the function *cor()* is used, but if it is selected *rcorr()* is used (which is part of the *Hmisc* package). The main implication is that *rcorr()* reports the results to only 2 decimal places (see the next section for a full description of these functions).

on Two-sided ◉. A one-tailed test should be selected when you have a directional hypothesis. With correlations, the direction of the relationship can be positive (e.g., 'the more anxious someone is about an exam, the better their mark will be') or negative (e.g., 'the more anxious someone is about an exam, the worse their mark will be'). A positive relationship means that the correlation coefficient will be greater than 0; therefore, if you predict a positive correlation coefficient then select Correlation < 0 ◉. However, if you predict a negative relationship then the correlation coefficient will be less than 0, so select Correlation < 0 ◉. For both the *correlation matrix* and *correlation test* dialog boxes click on OK to generate the output.

6.5.3. General procedure for correlations using R ①

To compute basic correlation coefficients there are three main functions that can be used: **cor()**, **cor.test()** and **rcorr()**. Table 6.2 shows the main differences between the three functions. The functions *cor()* and *cor.test()* are part of the base system in R, but *rcorr()* is part of the *Hmisc* package, so make sure you have it loaded.

Table 6.2 should help you to decide which function is best in a particular situation: if you want a confidence interval then you will have to use *cor.test()*, and if you want correlation coefficients for multiple pairs of variables then you cannot use *cor.test()*; similarly, if you want *p*-values then *cor()* won't help you. You get the gist.

Table 6.2 Attributes of different functions for obtaining correlations

Function	*Pearson*	*Spearman*	*Kendall*	*p-values*	*CI*	*Multiple Correlations?*	*Comments*
cor()	✓	✓	✓			✓	
cor.test()	✓	✓	✓	✓	✓		
rcorr()	✓	✓		✓		✓	2 d.p. only

We will look at each function in turn and see what parameters it uses. Let's start with *cor()*, which takes the general form:

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

in which:

- *x* is a numeric variable or dataframe.
- *y* is another numeric variable (does not need to be specified if *x* above is a dataframe).
- *use* is set equal to a character string that specifies how missing values are handled. The strings can be: (1) "everything", which will mean that R will output an NA instead of a correlation coefficient for any correlations involving variables containing missing values; (2) "all.obs", which will use all observations and, therefore, returns an error message if there are any missing values in the data; (3) "complete.obs", in which correlations are computed from only cases that are complete for all variables – sometimes known as *excluding cases listwise* (see R's Souls' Tip 6.1); or (4) "pairwise.complete.obs", in which correlations between pairs of variables are computed for cases that are complete for those two variables – sometimes known as *excluding cases pairwise* (see R's Souls' Tip 6.1).

- *method* enables you to specify whether you want "pearson", "spearman" or "kendall" correlations (note that all are written in lower case). If you want more than one type you can specify a list using the *c()* function; for example, *c("pearson", "spearman")* would produce both types of correlation coefficients.

If we stick with our exam anxiety data, then we could get Pearson correlations between all variables by specifying the dataframe (*examData*):

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

If we want a single correlation between a pair of variables (e.g., **Exam** and **Anxiety**) then we'd specify both variables instead of the dataframe:

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

We can get a different type of correlation (e.g., Kendall's tau) by changing the *method* command:

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

We can also change how we deal with missing values, for example, by asking for pairwise exclusion:

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

R's Souls' Tip 6.1 Exclude cases pairwise or listwise? ①

As we discover various functions in this book, many of them have options that determine how missing data are handled. Sometimes we can decide to exclude cases 'pairwise' or 'listwise'. Listwise means that if a case has a missing value for any variable, then they are excluded from the whole analysis. So, for example, in our exam anxiety data if one of our students had reported their anxiety and we knew their exam performance but we didn't have data about their revision time, then their data would not be used to calculate any of the correlations: *they would be completely excluded from the analysis*. Another option is to exclude cases on a pairwise basis, which means that if a participant has a score missing for a particular variable or analysis, then their data are excluded only from calculations involving the variable for which they have no score. For our student about whom we don't have any revision data, this means that their data would be excluded when calculating the correlation between exam scores and revision time, and when calculating the correlation between exam anxiety and revision time; however, the student's scores would be *included* when calculating the correlation between exam anxiety and exam performance because for this pair of variables we have both of their scores.

The function *rcorr()* is fairly similar to *cor()*. It takes the general form:

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

in which:

- *x* is a numeric variable or matrix.
- *y* is another numeric variable (does not need to be specified if *x* above is a matrix).
- *type* enables you to specify whether you want "pearson" or "spearman" correlations. If you want both you can specify a list as *c("pearson", "spearman")*.

A couple of things to note: first, this function does not work on dataframes, so you have to convert your dataframe to a matrix first (see section 3.9.2); second, this function excludes cases pairwise (see R's Souls' Tip 6.1) and there is no way to change this setting. Therefore, if you have two numeric variables (that are not part of a dataframe) called **Exam** and **Anxiety** then you could compute the Pearson correlation coefficient and its *p*-value by executing:

```
rcorr(Exam, Anxiety, type = "pearson")
```

Similarly, you could compute Pearson correlations (and their *p*-values) between all variables in a matrix called *examData* by executing:

```
rcorr(examData, type = "pearson")
```

The function *cor.test()* can be used only on pairs of variables (not a whole dataframe) and takes the general form:

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

in which:

- *x* is a numeric variable.
- *y* is another numeric variable.
- *alternative* specifies whether you want to do a two-tailed test (*alternative* = *"two.sided"*), which is the default, or whether you predict that the correlation will be less than zero (i.e., negative) or more than zero (i.e., positive), in which case you can use *alternative* = *"less"* and *alternative* = *"greater"*, respectively.
- *method* is the same as for *cor()* described above.
- *conf.level* allows you to specify the width of the confidence interval computed for the correlation. The default is 0.95 (*conf.level* = *0.95*) and if this is what you want then you don't need to use this command, but if you wanted a 90% or 99% confidence interval you could use *conf.level* = *0.9* and *conf.level* = *0.99*, respectively. Confidence intervals are produced only for Pearson's correlation coefficient.

Using our exam anxiety data, if we want a single correlation coefficient, its two-tailed *p*-value and 95% confidence interval between a pair of variables (for example, **Exam** and **Anxiety**) then we'd specify it much like we did for *cor()*:

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

If we predicted a negative correlation then we could add in the *alternative* command:

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

We could also specify a different confidence interval than 95%:

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

Hopefully you get the general idea. We will now move on to look at some examples of specific types of correlation coefficients.

OLIVER TWISTED

Please Sir, can I have some more … variance and covariance?

Oliver is so excited to get onto analysing his data that he doesn't want me to spend pages waffling on about variance and covariance. 'Stop writing, you waffling fool,' he says. 'I want to analyse my data.' Well, he's got a point. If you want to find out more about two functions for calculating variances and covariances that are part of the *cor()* family, then the additional material for this chapter on the companion website will tell you.

6.3.4. Pearson's correlation coefficient ①

6.5.4.1. Assumptions of Pearson's *r* ①

Pearson's (Figure 6.5) correlation coefficient was described in full at the beginning of this chapter. Pearson's correlation requires only that data are interval (see section 1.5.1.2) for it to be an accurate measure of the linear relationship between two variables. However, if you want to establish whether the correlation coefficient is significant, then more assumptions are required: for the test statistic to be valid the sampling distribution has to be normally distributed and as we saw in Chapter 5 we assume that it is if our sample data are normally distributed (or if we have a large sample). Although typically, to assume that the sampling distribution is normal, we would want both variables to be normally distributed, there is one exception to this rule: one of the variables can be a categorical variable provided there are only two categories (in fact, if you look at section 6.5.7 you'll see that this is the same as doing a *t*-test, but I'm jumping the gun a bit). In any case, if your data are non-normal (see Chapter 5) or are not measured at the interval level then you should use a different kind of correlation coefficient or use bootstrapping.

FIGURE 6.5
Karl Pearson

6.5.4.2. Computing Pearson's *r* using R ①

That's a confusing title. We have already gone through the nuts and bolts of using R Commander and the command line to calculate Pearson's *r*. We're going to use the exam anxiety data to get some hands-on practice.

SELF-TEST

✓ Load the **Exam Anxiety.dat** file into a dataframe called *examData*.

Let's look at a sample of this dataframe:

```
   Code Revise Exam Anxiety Gender
1     1      4   40  86.298   Male
2     2     11   65  88.716 Female
3     3     27   80  70.178   Male
4     4     53   80  61.312   Male
5     5      4   40  89.522   Male
6     6     22   70  60.506 Female
7     7     16   20  81.462 Female
8     8     21   55  75.820 Female
9     9     25   50  69.372 Female
10   10     18   40  82.268 Female
```

The first issue we have is that some of the variables are not numeric (**Gender**) and others are not meaningful numerically (**code**). We have two choices here. The first is to make a new dataframe by selecting only the variables of interest) – we discovered how to do this in section 3.9.1. The second is to specify this subset within the *cor()* command itself. If we choose the first method then we should execute:

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

The first line creates a dataframe (*examData2*) that contains all of the cases, but only the variables **Exam**, **Anxiety** and **Revise**. The second command creates a table of Pearson correlations between these three variables (note that Pearson is the default so we don't need to specify it and because there are no missing cases we do not need the *use* command).

Alternatively, we could specify the subset of variables in the *examData* dataframe as part of the *cor()* function:

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

The end result is the same, so it's purely down to preference. With the first method it is a little easier to see what's going on, but as you gain confidence and experience you might find that you prefer to save time and use the second method.

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

Output 6.1: Output for a Pearson's correlation

Output 6.1 provides a matrix of the correlation coefficients for the three variables. Each variable is perfectly correlated with itself (obviously) and so $r = 1$ along the diagonal of the table. Exam performance is negatively related to exam anxiety with a Pearson correlation coefficient of $r = -.441$. This is a reasonably big effect. Exam performance is positively related to the amount of time spent revising, with a coefficient of $r = .397$, which is also a reasonably big effect. Finally, exam anxiety appears to be negatively related to the time spent revising, $r = -.709$, which is a substantial effect size. In psychological terms, this all means that as anxiety about an exam increases, the percentage mark obtained in that exam decreases. Conversely, as the amount of time revising increases, the percentage obtained in the exam increases. Finally, as revision time increases, the student's anxiety about the exam decreases. So there is a complex interrelationship between the three variables.

Correlation coefficients are effect sizes, so we can interpret these values without really needing to worry about *p*-values (and as I have tried to drum into you, because *p*-values are related to sample size, there is a lot to be said for not obsessing about them). However, if you are the type of person who obsesses about *p*-values, then you can use the *rcorr()*

function instead and *p* yourself with excitement at the output it produces. First, make sure you have loaded the *Hmisc* package by executing:

```
library(Hmisc)
```

Next, we need to convert our dataframe into a matrix using the *as.matrix()* command. We can include only numeric variables so, just as we did above, we need to select only the numeric variables within the *examData* dataframe. To do this, execute:

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

Which creates a matrix called *examMatrix* that contains only the variables **Exam**, **Anxiety**, and **Revise** from the *examData* dataframe. To get the correlation matrix we simply input this matrix into the *rcorr()* function:[4]

```
rcorr(examMatrix)
```

As before, I think that the method above makes it clear what we're doing, but more experienced users could combine the previous two commands into a single one:

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

Output 6.2 shows the same correlation matrix as Output 6.1, except rounded to 2 decimal places. In addition, we are given the sample size on which these correlations are based, and also a matrix of *p*-values that corresponds to the matrix of correlation coefficients above. Exam performance is negatively related to exam anxiety with a Pearson correlation coefficient of $r = -.44$ and the significance value is less than .001 (it is approximately zero). This significance value tells us that the probability of getting a correlation coefficient this big in a sample of 103 people if the null hypothesis were true (there was no relationship between these variables) is very low (close to zero in fact). Hence, we can gain confidence that there is a genuine relationship between exam performance and anxiety. Our criterion for significance is usually .05 (see section 2.6.1) so we can say that all of the correlation coefficients are significant.

```
         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
```

Output 6.2

It can also be very useful to look at confidence intervals for correlation coefficients. Sadly, we have to do this one at a time (we can't do it for a whole dataframe or matrix). Let's look at the correlation between exam performance (**Exam**) and exam anxiety (**Anxiety**). We can compute the confidence interval *using cor.test()* by executing:

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

[4] The *ggm* package also has a function called *rcorr()*, so if you have this package installed, R might use that function instead, which will produce something very unpleasant on your screen. If so, you need to put *Hmisc::* in front of the commands to make sure R uses *rcorr()* from the *Hmisc* package (R's Souls' Tip 3.4):

```
Hmisc::rcorr(examMatrix)
Hmisc::rcorr(as.matrix(examData[, c("Exam", "Anxiety", "Revise")]))
```

Note that we have specified only the variables because by default this function produces Pearson's *r* and a 95% confidence interval. Output 6.3 shows the resulting output; it reiterates that the Pearson correlation between exam performance and anxiety was –.441, but tells us that this was highly significantly different from zero, $t(101) = -4.94$, $p < .001$. Most important, the 95% confidence ranged from –.585 to – .271, which does not cross zero. This tells us that in all likelihood, the population or actual value of the correlation is negative, so we can be pretty content that exam anxiety and exam performance are, in reality, negatively related.

```
       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
```

Output 6.3

SELF-TEST

✓ Compute the confidence intervals for the relationships between the time spent revising (**Revise**) and both exam performance (**Exam**) and exam anxiety (**Anxiety**).

6.5.4.3. Using R^2 for interpretation ①

Although we cannot make direct conclusions about causality from a correlation, we can take the correlation coefficient a step further by squaring it. The correlation coefficient squared (known as the **coefficient of determination**, R^2) is a measure of the amount of variability in one variable that is shared by the other. For example, we may look at the relationship between exam anxiety and exam performance. Exam performances vary from person to person because of any number of factors (different ability, different levels of preparation and so on). If we add up all of this variability (rather like when we calculated the sum of squares in section 2.4.1) then we would have an estimate of how much variability exists in exam performances. We can then use R^2 to tell us how much of this variability is shared by exam anxiety. These two variables had a correlation of −0.4410 and so the value of R^2 will be $(-0.4410)^2 = 0.194$. This value tells us how much of the variability in exam performance is shared by exam anxiety.

If we convert this value into a percentage (multiply by 100) we can say that exam anxiety shares 19.4% of the variability in exam performance. So, although exam anxiety was highly correlated with exam performance, it can account for only 19.4% of variation in exam scores. To put this value into perspective, this leaves 80.6% of the variability still to be accounted for by other variables.

You'll often see people write things about R^2 that imply causality: they might write 'the variance in *y accounted for* by *x*', or 'the variation in one variable *explained* by the other'. However, although R^2 is an extremely useful measure of the substantive importance of an effect, it cannot be used to infer causal relationships. Exam anxiety might well share 19.4% of the variation in exam scores, but it does not necessarily cause this variation.

We can get **R** to compute the coefficient of determination by remembering that "^2" means 'squared' in **R**-speak. Therefore, for our *examData2* dataframe (see earlier) if we execute:

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

instead of:

```
cor(examData2)
```

then you will see be a matrix containing r^2 instead of r (Output 6.4).

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

Output 6.4

Note that for exam performance and anxiety the value is 0.194, which is what we calculated above. If you want these values expressed as a percentage then simply multiply by 100, so the command would become:

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

6.5.5. Spearman's correlation coefficient ①

Spearman's correlation coefficient (Spearman, 1910), r_s, is a non-parametric statistic and so can be used when the data have violated parametric assumptions such as non-normally distributed data (see Chapter 5). You'll sometimes hear the test referred to as Spearman's rho (pronounced 'row', as in 'row your boat gently down the stream'). Spearman's test works by first ranking the data (see section 15.4.1), and then applying Pearson's equation (equation (6.3)) to those ranks.

I was born in England, which has some bizarre traditions. One such oddity is the World's Biggest Liar competition held annually at the Santon Bridge Inn in Wasdale (in the Lake District). The contest honours a local publican, 'Auld Will Ritson', who in the nineteenth century was famous in the area for his far-fetched stories (one such tale being that Wasdale turnips were big enough to be hollowed out and used as garden sheds). Each year locals are encouraged to attempt to tell the biggest lie in the world (lawyers and politicians are apparently banned from the competition). Over the years there have been tales of mermaid farms, giant moles, and farting sheep blowing holes in the ozone layer. (I am thinking of entering next year and reading out some sections of this book.)

Imagine I wanted to test a theory that more creative people will be able to create taller tales. I gathered together 68 past contestants from this competition and asked them where they were placed in the competition (first, second, third, etc.) and also gave them a creativity questionnaire (maximum score 60). The position in the competition is an ordinal variable (see section 1.5.1.2) because the places are categories but have a meaningful order (first place is better than second place and so on). Therefore, Spearman's correlation coefficient should be used (Pearson's r requires interval or ratio data). The data for this study are in the file **The Biggest Liar.dat**. The data are in two columns: one labelled **Creativity** and one labelled **Position** (there's actually a third variable in there but we will ignore it for the time being). For the **Position** variable, each of the categories described above has been coded with a numerical value. First place has been coded with the value 1, with positions being labelled 2, 3 and so on.

The procedure for doing a Spearman correlation is the same as for a Pearson correlation except that we need to specify that we want a Spearman correlation instead of Pearson, which is done using *method* = *"spearman"* for *cor()* and *cor.test()*, and *type* = *"spearman"* for *rcorr()*. Let's load the data into a dataframe and then create a dataframe by executing:

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

or if you haven't set your working directory, execute this command and use the dialog box to select the file:

```
liarData = read.delim(file.choose(),  header = TRUE)
```

SELF-TEST

✓ See whether you can use what you have learned so far to compute a Spearman's correlation between **Position** and **Creativity**.

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

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

Note that we have simply specified the two variables of interest, and then set the method to be a Spearman correlation. The output of this command will be:

```
[1] -0.3732184
```

If we want a significance value for this correlation we could either use *rcorr()* by executing (remembering that we have to first convert the dataframe to a matrix):

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

or simply use *cor.test()*, which has the advantage that we can set a directional hypothesis. I predicted that more creative people would tell better lies. Doing well in the competition (i.e., telling better lies) actually equates to a lower number for the variable **Position** (first place = 1, second place = 2 etc.), so we're predicting a negative relationship. High scores on **Creativity** should equate to a lower value of **Position** (because a low value means you did well!). Therefore, we predict that the correlation will be less than zero, and we can reflect this prediction by using *alternative* = *"less"* in the command:

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

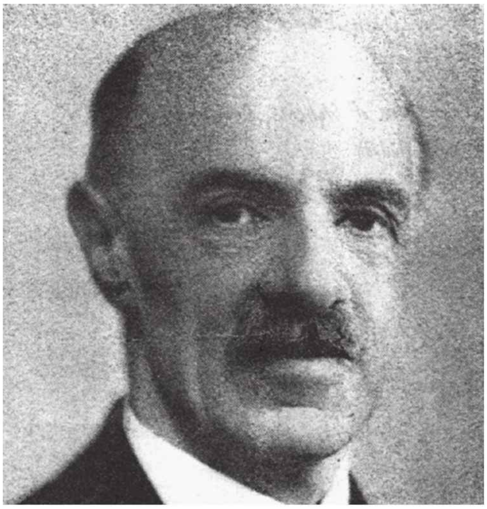


FIGURE 6.6 Charles Spearman, ranking furiously

```
              Spearman’s rank correlation rho
data:  liarData$Position and liarData$Creativity
S = 71948.4, p-value = 0.0008602
alternative hypothesis: true rho is less than 0
sample estimates:
       rho
-0.3732184
```

Output 6.5

Output 6.5 shows the output for a Spearman correlation on the variables **Creativity** and **Position**. The output is very similar to that of the Pearson correlation (except that confidence intervals are not produced – if you want one see the section on bootstrapping): the correlation coefficient between the two variables is fairly large (−.373), and the significance value of this coefficient is very small (p < .001). The significance value for this correlation coefficient is less than .05; 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. Note that the relationship is negative: as creativity increased, position decreased. Remember that a low number means that you did well in the competition (a low number such as 1 means you came first, and a high number like 4 means you came fourth). Therefore, our hypothesis is supported: as creativity increased, so did success in the competition.

SELF-TEST

✓ Did creativity cause success in the World’s Biggest Liar competition?

6.5.6. Kendall’s tau (non-parametric) ①

Kendall’s tau, τ, is another non-parametric correlation and it should be used rather than Spearman’s coefficient when you have a small data set with a large number of tied ranks. This means that if you rank all of the scores and many scores have the same rank, then Kendall’s tau should be used. Although Spearman’s statistic is the more popular of the two coefficients, there is much to suggest that Kendall’s statistic is actually a better estimate of the correlation in the population (see Howell, 1997: 293). As such, we can draw more accurate generalizations from Kendall’s statistic than from Spearman’s. To carry out Kendall’s correlation on the World’s Biggest Liar data simply follow the same steps as for Pearson and Spearman correlations but use *method = “kendall”*:

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

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

The output is much the same as for Spearman’s correlation.

```
       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
```

Output 6.6

You'll notice from Output 6.6 that the actual value of the correlation coefficient is closer to zero than the Spearman correlation (it has increased from −.373 to −.300). Despite the difference in the correlation coefficients we can still interpret this result as being a highly significant relationship (because the significance value of .001 is less than .05). However, Kendall's value is a more accurate gauge of what the correlation in the population would be. As with the Pearson correlation, we cannot assume that creativity caused success in the World's Best Liar competition.

SELF-TEST

✓ Conduct a Pearson correlation analysis of the advert data from the beginning of the chapter.

6.5.5. Bootstrapping correlations ③

Another way to deal with data that do not meet the assumptions of Pearson's *r* is to use bootstrapping. The *boot()* function takes the general form:

```
object<-boot(data, function, replications)
```

in which *data* specifies the dataframe to be used, *function* is a function that you write to tell *boot()* what you want to bootstrap, and *replications* is a number specifying how many bootstrap samples you want to take (I usually set this value to 2000). Executing this command creates an *object* that has various properties. We can view an estimate of bias, and an empirically derived standard error by viewing *object*, and we can display confidence intervals based on the bootstrap by executing *boot.ci(object)*.

When using the *boot()* function with correlations (and anything else for that matter) the tricky bit is writing the function (R's Souls' Tip 6.2). If we stick with our biggest liar data and want to bootstrap Kendall tau, then our function will be:

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

Executing this command creates an object called *bootTau*. The first bit of the function tells R what input to expect in the function: in this case we need to feed a dataframe (*liarData*) into the function and a variable that has been called *i* (which refers to a particular bootstrap sample). The second part of the function specifies the *cor()* function, which is the thing we want to bootstrap. Notice that *cor()* is specified in exactly the same way as when we did the original Kendall correlation except that for each variable we have added *[i]*, which again just refers to a particular bootstrap sample. If you want to bootstrap a Pearson or Spearman correlation you do it in exactly the same way except that you specify *method = "pearson"* or *method = "spearman"* when you define the function.

To create the bootstrap object, we execute:

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

The first command loads the *boot* package (in case you haven't already initiated it). The second command creates an object (*boot_kendall*) based on bootstrapping the *liarData* dataframe using the *bootTau* function that we previously defined and executed. The second line displays a summary of the *boot_kendall* object. To get the 95% confidence interval for the *boot_kendall* object we execute:[5]

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

Output 6.7 shows the contents of both *boot_kendall* and also the output of the *boot.ci()* function. First, we get the original value of Kendall's tau (–.300), which we computed in the previous section. We also get an estimate of the bias in that value (which in this case is very small) and the standard error (0.098) based on the bootstrap samples. The output from *boot.ci()* gives us four different confidence intervals (the basic bootstrapped CI, percentile and BCa). The good news is that none of these 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 conclusions stand.

```
ORDINARY NONPARAMETRIC BOOTSTRAP

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

Bootstrap Statistics :
      original      bias    std. error
t1* -0.3002413 0.001058191    0.097663


> boot.ci(boot_kendall)

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 2000 bootstrap replicates

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

Intervals :
Level      Normal              Basic
95%   (-0.4927, -0.1099 )   (-0.4956, -0.1126 )

Level     Percentile            BCa
95%   (-0.4879, -0.1049 )   (-0.4777, -0.0941 )
Calculations and Intervals on Original Scale
Warning message:
In boot.ci(boot_kendall) :
  bootstrap variances needed for studentized intervals
```

Output 6.7

SELF-TEST

✓ Conduct bootstrap analysis of the Pearson and Spearman correlations for the *examData2* dataframe.

[5] If we want something other than a 95% confidence interval we can add *conf* = *x*, in which *x* is the value of the confidence interval as a proportion. For example, we can get a 99% confidence interval by executing:

```
boot.ci(boot_kendall, conf = 0.99)
```

R's Souls' Tip 6.2 Writing functions ③

What happens if there is not a function available in **R** to do what you want to do? Simple, write your own function. The ability to write your own functions is a very powerful feature of **R**. With a sufficient grasp of the **R** environment (and the maths behind whatever you're trying to do) you can write a function to do virtually anything for you (apart from making coffee). To write a function you need to execute a string of commands that define the function. They take this general format:

```
nameofFunction<-function(inputObject1, inputObject2, etc.)
{
       a set of commands that do things to the input object(s)
       a set of commands that specify the output of the function
}
```

Basically, you name the function (any name you like, but obviously one that tells you what the function does is helpful). The *function()* tells **R** that you're writing a function, and you need to place within the brackets anything you want as input to the function: this can be any object in **R** (a model, a dataframe, a numeric value, text, etc.). A function might just accept one object, or there might be many. The names you list in the brackets can be whatever you like, but again it makes sense to label them based on what they are (e.g., if you need to input a dataframe then it makes sense to give the input a label of *dataframe* so that you remember what it is that the function needs). You then use {} to contain a set of instructions that tell **R** what to do with the objects that have been input into the function. These are usually some kind of calculations followed by some kind of instruction about what to return from the function (the output).

Imagine that **R** doesn't have a function for computing the mean and we wanted to write one (this will keep things familiar). We could write this as:

```
meanOfVariable<-function(variable)
{
       mean<-sum(variable)/length(variable)
       cat("Mean = ", mean)
}
```

Executing this command creates a function called *meanOfVariable* that expects a variable to be entered into it. The bits in {} tell **R** what to do with the variable that is entered into the function. The first line computes the mean using the function *sum()* to add the values in the variable that was entered into the function, and the function *length()* counts how many scores are in the variable. Therefore, *mean<-sum(variable)/length(variable)* translates as mean = sum of scores/number of scores (which, of course, is the definition of the mean). The final line uses the *cat()* function to print the text "Mean =" and the value of *mean* that we have just computed.

Remember the data about the number of friends that statistics lecturers had that we used to explore the mean in Chapter 2 (section 2.4.1). We could enter these data by executing:

```
lecturerFriends = c(1,2,3,3,4)
```

Having executed our function, we can use it to find the mean. We simply execute:

```
meanOfVariable(lecturerFriends)
```

This tells **R** that we want to use the function *meanOfVariable()*, which we have just created, and that the variable we want to apply this function to is **lecturerFriends**. Executing this command gives us:

```
Mean =  2.6
```

In other words, **R** has printed the text 'Mean =' and the value of the mean computed by the function (just as we asked it to). This value is the same as the one we calculated in section 2.4.1, so the function has worked. The beauty of functions is that having executed the commands that define it, we can use this function over and over again within our session (which saves time).

As a final point, just to be clear, when we define our function we can name things anything we like. For example, although I named the input to the function 'variable' to remind myself what the function needs, I could have named it 'HarryTheHungryHippo' if I had wanted to. Provided that I carry this name through to the commands within the function, it will work:

```
meanOfVariable<-function(HarryTheHungryHippo)
{
    mean<-sum(HarryTheHungryHippo)/length(HarryTheHungryHippo)
    cat("Mean = ", mean)
}
```

Note that within the function I now apply the *sum()* and *length()* functions to *HarryTheHungryHippo* because this is the name that I gave to the input of the function. It will work, but people will be probably confused about what *HarryTheHungryHippo* is when they read your code.

6.5.8. Biserial and point-biserial correlations ③

The biserial and point-biserial correlation coefficients are distinguished by only a conceptual difference, yet their statistical calculation is quite different. These correlation coefficients are used when one of the two variables is **dichotomous** (i.e., it is categorical with only two categories). An example of a dichotomous variable is being pregnant, because a woman can be either pregnant or not (she cannot be 'a bit pregnant'). Often it is necessary to investigate relationships between two variables when one of the variables is dichotomous. The difference between the use of biserial and point-biserial correlations depends on whether the dichotomous variable is discrete or continuous. This difference is very subtle. A discrete, or true, dichotomy is one for which there is no underlying continuum between the categories. An example of this is whether someone is dead or alive: a person can be only dead or alive, they can't be 'a bit dead'. Although you might describe a person as being 'half-dead' – especially after a heavy drinking session – they are clearly still alive if they are still breathing! Therefore, there is no continuum between the two categories. However, it is possible to have a dichotomy for which a continuum does exist. An example is passing or failing a statistics test: some people will only just fail while others will fail by a large margin; likewise some people will scrape a pass while others will excel. So although participants fall into only two categories there is an underlying continuum along which people lie. Hopefully, it is clear that in this case there is some kind of continuum underlying the dichotomy, because some people passed or failed more dramatically than others. The **point-biserial correlation** coefficient (r_{pb}) is used when one variable is a discrete dichotomy (e.g., pregnancy), whereas the **biserial correlation** coefficient (r_b) is used when one variable is a continuous dichotomy (e.g., passing or failing an exam).

Imagine that I was interested in the relationship between the gender of a cat and how much time it spent away from home (what can I say? I love cats so these things interest me). I had heard that male cats disappeared for substantial amounts of time on long-distance roams around the neighbourhood (something about hormones driving them to find mates) whereas female cats tended to be more homebound. So, I used this as a purr-fect (sorry!) excuse to go and visit lots of my friends and their cats. I took a note of the gender of the cat and then asked the owners to note down the number of hours that their cat was absent from home over a week. Clearly the time spent away from home is measured at an interval level – and let's assume it meets the other assumptions of parametric data – while the gender of the cat is discrete dichotomy. A point-biserial correlation has to be calculated and this is simply a Pearson correlation when the dichotomous variable is coded with 0 for one category and 1 for the other.

Let's load the data in the file **pbcorr.csv** and have a look at it. These data are in the CSV format, so we can load them as (assuming you have set the working directory correctly):

```
catData = read.csv("pbcorr.csv",  header = TRUE)
```

Note that we have used the *read.csv()* function because the file is a .csv file. To look at the data execute:

```
catData
```

A sample of the data is as follows:

```
   time gender recode
1    41      1      0
2    40      0      1
3    40      1      0
4    38      1      0
5    34      1      0
6    46      0      1
7    42      1      0
8    42      1      0
9    47      1      0
10   42      0      1
11   45      1      0
12   46      1      0
13   44      1      0
14   54      0      1
```

There are three variables:

- **time**, which is the number of hours that the cat spent away from home (in a week).
- **gender**, is the gender of the cat, coded as 1 for male and 0 for female.
- **recode**, is the gender of the cat but coded the opposite way around (i.e., 0 for male and 1 for female). We will come to this variable later, but for now ignore it.

SELF-TEST

✓ Carry out a Pearson correlation on **time** and **gender**.

Congratulations: if you did the self-test task then you have just conducted your first point-biserial correlation. See, despite the horrible name, it's really quite easy to do. If you didn't do the self-test then execute:

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

You should find that you can see Output 6.8. The point-biserial correlation coefficient is r_{pb} = .378, which has a significance value of .003. The significance test for this correlation is actually the same as performing an independent-samples *t*-test on the data (see Chapter 9). The sign of the correlation (i.e., whether the relationship was positive or negative) will depend entirely on which way round the coding of the dichotomous variable was made. To prove that this is the case, the data file **pbcorr.dat** has an extra variable called **recode** which is the same as the variable **gender** except that the coding is reversed (1 = female, 0 = male). If you repeat the Pearson correlation using **recode** instead of **gender** you will find that the correlation coefficient becomes −.378. The sign of the coefficient is completely dependent on which category you assign to which code and so we must ignore all information about the direction of the relationship. However, we can still interpret R^2 as before. So in this example, $R^2 = .378^2 = .143$. Hence, we can conclude that gender accounts for 14.3% of the variability in time spent away from home.

SELF-TEST

✓ Carry out a Pearson correlation on **time** and **recode**.

```
        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
```

Output 6.8

Imagine now that we wanted to convert the point-biserial correlation into the biserial correlation coefficient (r_b) (because some of the male cats were neutered and so there might be a continuum of maleness that underlies the gender variable). We must use equation (6.9) in which p is the proportion of cases that fell into the largest category and q is the proportion of cases that fell into the smallest category. Therefore, p and q are simply the number of male and female cats. In this equation y is the ordinate of the normal distribution at the point where there is p% of the area on one side and q% on the other (this will become clearer as we do an example):

$$r_b = \frac{r_{pb}\sqrt{pq}}{y} \tag{6.9}$$

To calculate p and q, we first need to use the **table()** function to compute the frequencies of males and female cats. We will store these frequencies in a new object called *catFrequencies*. We then use this object to compute the proportion of male and female cats using the **prop.table()** function. We execute these two commands as follows:

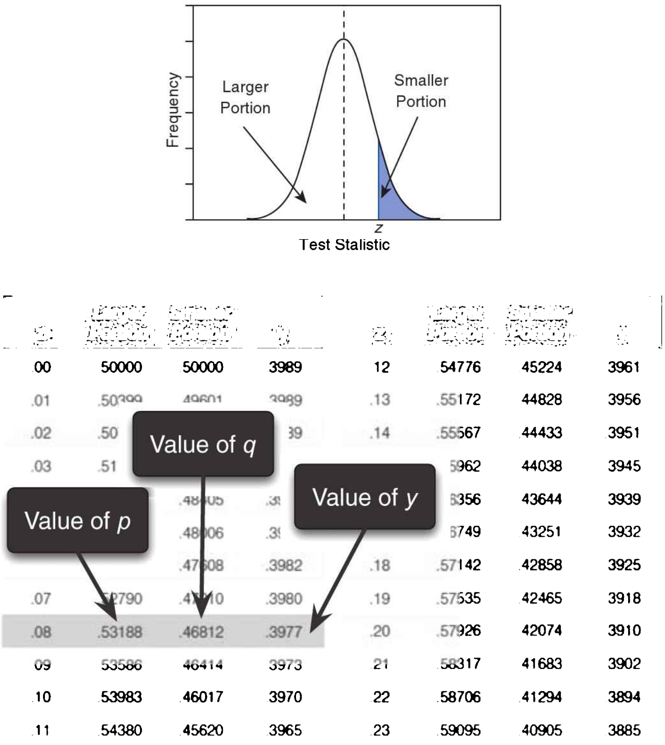
```
catFrequencies<-table(catData$gender)
prop.table(catFrequencies)
```

The resulting output tells us that the proportion of male cats (1) was .467 (this is q because it is the smallest portion) and the proportion of females (0) was .533 (this is p because it is the largest portion):

```
        0         1
0.5333333 0.4666667
```

To calculate y, we use these values and the values of the normal distribution displayed in the Appendix. Figure 6.7 shows how to find the ordinate (the value in the column labelled y)

when the normal curve is split with .467 as the smaller portion and .533 as the larger portion. The figure shows which columns represent p and q and we look for our values in these columns (the exact values of 0.533 and 0.467 are not in the table so instead we use the nearest values that we can find, which are .5319 and .4681, respectively). The ordinate value is in the column y and is .3977.



[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
.00	50000	50000	3989	12	.54776	45224	3961
.01	.50399	.49601	.3989	.13	.55172	44828	3956
.02	[illegible]	[illegible]	[illegible]	.14	.55567	44433	3951
.03	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	44038	3945
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	43644	3939
[illegible]	[illegible]	.48006	[illegible]	[illegible]	[illegible]	43251	3932
[illegible]	[illegible]	.47608	.3982	.18	.57142	42858	3925
.07	[illegible]	[illegible]	.3980	.19	.57535	42465	3918
.08	.53188	.46812	.3977	.20	.57926	42074	3910
.09	.53586	.46414	.3973	.21	.58317	41683	3902
.10	.53983	.46017	.3970	22	.58706	.41294	3894
.11	.54380	.45620	.3965	23	.59095	40905	3885

FIGURE 6.7 Getting the 'ordinate' of the normal distribution

If we replace these values in equation (6.9) we get .475 (see below), which is quite a lot higher than the value of the point-biserial correlation (0.378). This finding just shows you that whether you assume an underlying continuum or not can make a big difference to the size of effect that you get:

$$r_b = \frac{r_{pb}\sqrt{pq}}{y} = \frac{.378\sqrt{.533 \times .467}}{.3977} = .475$$

If this process freaks you out, then luckily you can get **R** to do it for you by installing the *polycor* package and using the *polyserial()* function. You can simply specify the two variables of interest within this function just as you have been doing for every other correlation in this chapter. Execute this command:

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

and the resulting output:

```
[1] 0.4749256
```

confirms out earlier calculation.

You might wonder, given that you can get **R** to calculate the biserial correlation in one line of code, why I got you to calculate it by hand. It's entirely plausible that I'm just a nasty person who enjoys other people's pain. An alternative explanation is that the values of p and q are about to come in handy so it was helpful to show you how to calculate them. I'll leave you to decide which explanation is most likely.

To get the significance of the biserial correlation we need to first work out its standard error. If we assume the null hypothesis (that the biserial correlation in the population is zero) then the standard error is given by (Terrell, 1982):

$$SE_{r_b} = \frac{\sqrt{pq}}{y\sqrt{N}} \tag{6.10}$$

This equation is fairly straightforward because it uses the values of p, q and y that we already used to calculate the biserial r. The only additional value is the sample size (N), which in this example was 60. So our standard error is:

$$SE_{r_b} = \frac{\sqrt{.533 \times .467}}{.3977 \times \sqrt{60}} = .162$$

The standard error helps us because we can create a z-score (see section 1.7.4). To get a z-score we take the biserial correlation, subtract the mean in the population and divide by the standard error. We have assumed that the mean in the population is 0 (the null hypothesis), so we can simply divide the biserial correlation by its standard error:

$$z_{r_b} = \frac{r_b - \bar{r}_b}{SE_{r_b}} = \frac{r_b - 0}{SE_{r_b}} = \frac{r_b}{SE_{r_b}} = \frac{.475}{.162} = 2.93$$

We can look up this value of z (2.93) in the table for the normal distribution in the Appendix and get the one-tailed probability from the column labelled 'Smaller Portion'. In this case the value is .00169. To get the two-tailed probability we simply multiply the one-tailed probability value by 2, which gives us .00338. As such the correlation is significant, $p < .01$.

CRAMMING SAM'S TIPS Correlaion coefficients

- We can measure the relationship between two variables using *correlation coefficients*.
- These coefficients lie between −1 and +1.
- *Pearson's correlation coefficient*, r, is a parametric statistic and requires interval data for both variables. To test its significance we assume normality too.
- *Spearman's correlation coefficient*, r_s, is a non-parametric statistic and requires only ordinal data for both variables.
- *Kendall's correlation coefficient*, τ, is like Spearman's r_s but probably better for small samples.
- The *point-biserial correlation coefficient*, r_{pb}, quantifies the relationship between a continuous variable and a variable that is a discrete dichotomy (e.g., there is no continuum underlying the two categories, such as dead or alive).
- The *biserial correlation coefficient*, r_b, quantifies the relationship between a continuous variable and a variable that is a continuous dichotomy (e.g., there is a continuum underlying the two categories, such as passing or failing an exam).

6.6. Partial correlation ②

6.6.1. The theory behind part and partial correlation ②

I mentioned earlier that there is a type of correlation that can be done that allows you to look at the relationship between two variables when the effects of a third variable are held constant. For example, analyses of the exam anxiety data (in the file **Exam Anxiety.dat**) showed that exam performance was negatively related to exam anxiety, but positively related to revision time, and revision time itself was negatively related to exam anxiety. This scenario is complex, but given that we know that revision time is related to both exam anxiety and exam performance, then if we want a pure measure of the relationship between exam anxiety and exam performance we need to take account of the influence of revision time. Using the values of R^2 for these relationships (refer back to Output 6.4), we know that exam anxiety accounts for 19.4% of the variance in exam performance, that revision time accounts for 15.7% of the variance in exam performance, and that revision time accounts for 50.2% of the variance in exam anxiety. If revision time accounts for half of the variance in exam anxiety, then it seems feasible that at least some of the 19.4% of variance in exam performance that is accounted for by anxiety is the same variance that is accounted for by revision time. As such, some of the variance in exam performance explained by exam anxiety is not ***unique*** and can be accounted for by revision time. A correlation between two variables in which the effects of other variables are held constant is known as a **partial correlation.**

Let's return to our example of exam scores, revision time and exam anxiety to illustrate the principle behind partial correlation (Figure 6.8). In part 1 of the diagram there is a box for exam performance that represents the total variation in exam scores (this value would be the variance of exam performance). There is also a box that represents the variation in exam anxiety (again, this is the variance of that variable). We know already that exam anxiety and exam performance share 19.4% of their variation (this value is the correlation coefficient squared). Therefore, the variations of these two variables overlap (because they share variance) creating a third box (the blue cross hatched box). The overlap of the boxes representing exam performance and exam anxiety is the common variance. Likewise, in part 2 of the diagram the shared variation between exam performance and revision time is illustrated. Revision time shares 15.7% of the variation in exam scores. This shared variation is represented by the area of overlap (the dotted-blue lines box). We know that revision time and exam anxiety also share 50% of their variation; therefore, it is very probable that some of the variation in exam performance shared by exam anxiety is the same as the variance shared by revision time.

Part 3 of the diagram shows the complete picture. The first thing to note is that the boxes representing exam anxiety and revision time have a large overlap (this is because they share 50% of their variation). More important, when we look at how revision time and anxiety contribute to exam performance we see that there is a portion of exam performance that is shared by both anxiety and revision time (the white area). However, there are still small chunks of the variance in exam performance that are unique to the other two variables. So, although in part 1 exam anxiety shared a large chunk of variation in exam performance, some of this overlap is also shared by revision time. If we remove the portion of variation that is also shared by revision time, we get a measure of the unique relationship between exam performance and exam anxiety. We use partial correlations to find out the size of the unique portion of variance. Therefore, we could conduct a partial correlation between exam anxiety and exam performance while 'controlling' for the effect of revision time. Likewise, we could carry out a partial correlation between revision time and exam performance while 'controlling' for the effects of exam anxiety.

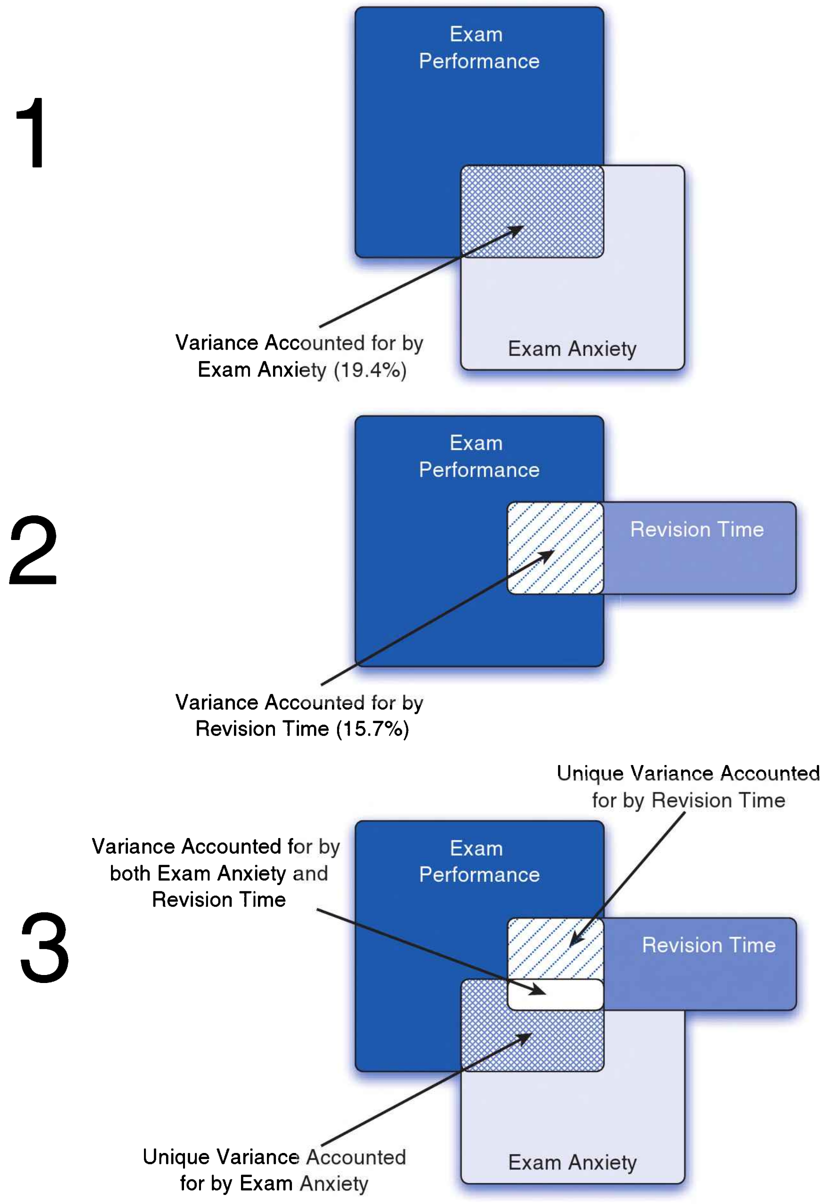

1
Exam Performance
Exam Anxiety
Variance Accounted for by Exam Anxiety (19.4%)
2
Exam Performance
Revision Time
Variance Accounted for by Revision Time (15.7%)
3
Unique Variance Accounted for by Revision Time
Variance Accounted for by both Exam Anxiety and Revision Time
Exam Performance
Revision Time
Exam Anxiety
Unique Variance Accounted for by Exam Anxiety

FIGURE 6.8 Diagram showing the principle of partial correlation

6.6.2. Partial correlation using R ②

We will use the *examData2* dataframe again, so if you haven't got this loaded then execute these commands:

```
examData = read.delim("Exam Anxiety.dat",  header = TRUE)
examData2 <- examData[, c("Exam", "Anxiety", "Revise")]
```

This will import the **Exam Anxiety.dat** file and create a dataframe containing only the three variables of interest. 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. These are part of the *ggm* package, so first load this:

```
library(ggm)
```

The general form of *pcor()* is:

```
pcor(c("var1", "var2", "control1", "control2" etc.), var(dataframe))
```

Basically, you enter a list of variables as strings (note the variable names have to be in quotes) using the *c()* function. The first two variables should be those for which you want the partial correlation; any others listed should be variables for which you'd like to 'control'. You can 'control' for the effects of a single variable, in which case the resulting coefficient is known as a *first-order partial correlation*; it is also possible to control for the effects of two (a *second-order partial correlation*), three (a *third-order partial correlation*), or more variables at the same time. The second part of the function simply asks for the name of the dataframe (in this case *examData2*). For the current example, we want the correlation between exam anxiety and exam performance (so we list these variables first) controlling for exam revision (so we list this variable afterwards). As such, we can execute the following command:

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

Executing this command will print the partial correlation to the console. However, I find it useful to create an object containing the partial correlation value so that we can use it in other commands. As such, I suggest that you execute this command to create an object called *pc*:

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

We can then see the partial correlation and the value of R^2 in the console by executing:

```
pc
pc^2
```

The general form of *pcor.test()* is:

```
pcor(pcor object, number of control variables, sample size)
```

Basically, you enter an object that you have created with *pcor()* (or you can put the *pcor()* command directly into the function). 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)
```

to see the significance of the partial correlation.

```
> pc
[1] -0.2466658
> pc^2
[1] 0.06084403
> pcor.test(pc, 1, 103)
$tval
[1] -2.545307


$df
[1] 100

$pvalue
[1] 0.01244581
```

Output 6.9

Output 6.9 shows the output for the partial correlation of exam anxiety and exam performance controlling for revision time; it also shows the squared value that we calculated (*pc* ^ 2), and the significance value obtained from *pcor.test()*. The output of *pcor()* is the partial correlation for the variables **Anxiety** and **Exam** but controlling for the effect of **Revision**. First, notice that 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 ($r = -.441$). In fact, the correlation coefficient is nearly half what it was before. Although this correlation is still statistically significant (its *p*-value is .012, which is still below .05), the relationship is diminished. In terms of variance, the value of R^2 for the partial correlation is .06, which means that exam anxiety can now account for only 6% of the variance in exam performance. When the effects of revision time were not controlled for, exam anxiety shared 19.4% of the variation in exam scores and so the inclusion of revision time has severely diminished the amount of variation in exam scores shared by anxiety. As such, a truer measure of the role of exam anxiety has been obtained. Running this analysis has shown us that exam anxiety alone does explain some of the variation in exam scores, but there is a complex relationship between anxiety, revision and exam performance that might otherwise have been ignored. Although causality is still not certain, because relevant variables are being included, the third variable problem is, at least, being addressed in some form.

These partial correlations can be done when variables are dichotomous (including the 'third' variable). So, for example, we could look at the relationship between bladder relaxation (did the person wet themselves or not?) and the number of large tarantulas crawling up your leg, controlling for fear of spiders (the first variable is dichotomous, but the second variable and 'controlled for' variables are continuous). Also, to use an earlier example, we could examine the relationship between creativity and success in the World's Biggest Liar competition, controlling for whether someone had previous experience in the competition (and therefore had some idea of the type of tale that would win) or not. In this latter case the 'controlled for' variable is dichotomous.[6]

6.6.2. Semi-partial (or part) correlations ②

In the next chapter, we will come across another form of correlation known as a **semi-partial correlation** (also referred to as a **part correlation**). While I'm babbling on about partial correlations it is worth my explaining the difference between this type of correlation and semi-partial correlation. When we do a partial correlation between two variables, we control for the effects of a third variable. Specifically, the effect that the third variable has on *both* variables in the correlation is controlled. In a semi-partial correlation we control for the effect that the third variable has on only one of the variables in the correlation. Figure 6.9 illustrates this principle for the exam performance data. The partial correlation that we

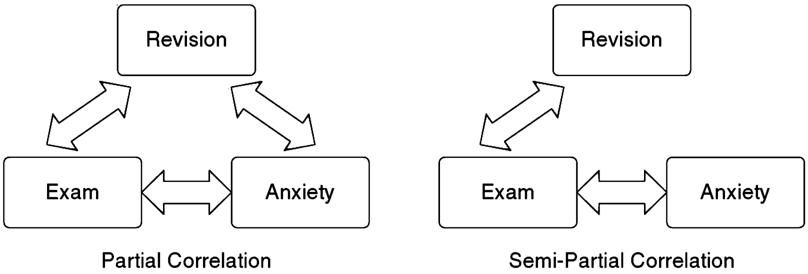


FIGURE 6.9 The difference between a partial and a semi-partial correlation

[6] Both these examples are, in fact, simple cases of hierarchical regression (see the next chapter) and the first example is also an example of analysis of covariance. This may be confusing now, but as we progress through the book I hope it'll become clearer that virtually all of the statistics that you use are actually the same things dressed up in different names.

calculated took account not only of the effect of revision on exam performance, but also of the effect of revision on anxiety. If we were to calculate the semi-partial correlation for the same data, then this would control for only the effect of revision on exam performance (the effect of revision on exam anxiety is ignored). Partial correlations are most useful for looking at the unique relationship between two variables when other variables are ruled out. Semi-partial correlations are, therefore, useful when trying to explain the variance in one particular variable (an outcome) from a set of predictor variables. (Bear this in mind when you read Chapter 7.)

CRAMMING SAM'S TIPS

Partial and semi-partial correlation

- A *partial correlation* quantifies the relationship between two variables while controlling for the effects of a third variable on *both* variables in the original correlation.
- A *semi-partial correlation* quantifies the relationship between two variables while controlling for the effects of a third variable on only *one* of the variables in the original correlation.

6.7. Comparing correlations ③

6.7.1. Comparing independent *r*s ③

Sometimes we want to know whether one correlation coefficient is bigger than another. For example, when we looked at the effect of exam anxiety on exam performance, we might have been interested to know whether this correlation was different in men and women. We could compute the correlation in these two samples, but then how would we assess whether the difference was meaningful?

SELF-TEST

✓ Use the *subset()* function to compute the correlation coefficient between exam anxiety and exam performance in men and women.

If we did this, we would find that the correlations were $r_{\text{Male}} = -.506$ and $r_{\text{Female}} = -.381$. These two samples are independent; that is, they contain different entities. To compare these correlations we can again use what we discovered in section 6.3.3 to convert these coefficients to z_r (just to remind you, we do this because it makes the sampling distribution normal and, therefore, we know the standard error). If we do the conversion, then we get z_r (males) = −.557 and z_r (females) = −.401. We can calculate a z-score of the differences between these correlations as:

$$z_{\text{Difference}} = \frac{z_{r_1} - z_{r_2}}{\sqrt{\frac{1}{N_1 - 3} + \frac{1}{\sqrt{N_2 - 3}}}} \tag{6.11}$$

We had 52 men and 51 women so we would get:

$$z_{\text{Difference}} = \frac{-.557-(-.401)}{\sqrt{\frac{1}{49}+\frac{1}{48}}} = \frac{-.156}{0.203} = -\ 0.768$$

We can look up this value of z (0.768; we can ignore the minus sign) in the table for the normal distribution in the Appendix and get the one-tailed probability from the column labelled 'Smaller Portion'. In this case the value is .221. To get the two-tailed probability we simply multiply the one-tailed probability value by 2, which gives us .442. As such the correlation between exam anxiety and exam performance is not significantly different in men and women (see Oliver Twisted for how to do this using **R**).

OLIVER TWISTED

Please Sir, can I have some more … functions?

'These equations are rubbish,' says Oliver, 'they're too confusing and I hate them. Can't we get **R** to do it for us while we check Facebook?' Well, no, you can't. Except you sort of can by writing your own function. 'Write my own function!!' screams Oliver whilst trying to ram his computer keyboard into his mouth. 'You've got to be joking, you steaming dog colon, I can barely write my own name.' Luckily for you Oliver, I've done it for you. To find out more, read the additional material for this chapter on the companion website. Or check Facebook, the choice is yours.

6.6.2. Comparing dependent *r*s ③

If you want to compare correlation coefficients that come from the same entities then things are a little more complicated. You can use a t-statistic to test whether a difference between two dependent correlations from the same sample is significant. For example, in our exam anxiety data we might want to see whether the relationship between exam anxiety (x) and exam performance (y) is stronger than the relationship between revision (z) and exam performance. To calculate this, all we need are the three rs that quantify the relationships between these variables: r_{xy}, the relationship between exam anxiety and exam performance (−.441); r_{zy}, the relationship between revision and exam performance (.397); and r_{xz}, the relationship between exam anxiety and revision (−.709). The t-statistic is computed as (Chen & Popovich, 2002):

$$t_{\text{Difference}} = (r_{xy} - r_{zy})\sqrt{\frac{(n-3)(1+r_{xz})}{2(1-r_{xy}^2 - r_{xz}^2 - r_{zy}^2 + 2r_{xy}r_{xz}r_{zy}}} \quad (6.12)$$

Admittedly that equation looks hideous, but really it's not too bad: it just uses the three correlation coefficients and the sample size N.

Put in the numbers from the exam anxiety example (N was 103) and you should end up with:

$$t_{\text{Difference}} = (-.838)\sqrt{\frac{29.1}{2(1-.194-.503-.158+0.248)}} = -5.09$$

This value can be checked against the appropriate critical value in the Appendix with $N-3$ degrees of freedom (in this case 100). The critical values in the table are 1.98 ($p < .05$) and 2.63 ($p < .01$), two-tailed. As such we can say that the correlation between exam anxiety and exam performance was significantly higher than the correlation between revision time and exam performance (this isn't a massive surprise, given that these relationships went in the opposite directions to each other).

OLIVER TWISTED

Please Sir, can I have some more … comparing of correlations?

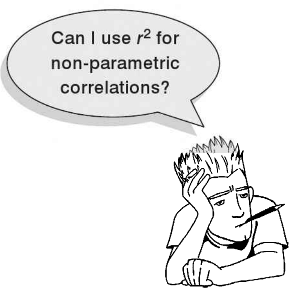
'Are you having a bloody laugh with that equation?' yelps Oliver. 'I'd rather smother myself with cheese sauce and lock myself in a room of hungry mice.' Yes, yes, Oliver, enough of your sexual habits. To spare the poor mice I have written another **R** function to run the comparison mentioned in this section. For a guide on how to use them read the additional material for this chapter on the companion website. Go on, be kind to the mice!

6.8. Calculating the effect size ①

Calculating effect sizes for correlation coefficients couldn't be easier because, as we saw earlier in the book, correlation coefficients *are* effect sizes! So, no calculations (other than those you have already done) necessary! However, I do want to point out one caveat when using non-parametric correlation coefficients as effect sizes. Although the Spearman and Kendall correlations are comparable in many respects (their power, for example, is similar under parametric conditions), there are two important differences (Strahan, 1982).

First, we saw for Pearson's r that we can square this value to get the proportion of shared variance, R^2. For Spearman's r_s we can do this too because it uses the same equation as Pearson's r. However, the resulting R_s^2 needs to be interpreted slightly differently: it is the proportion of variance in the *ranks* that two variables share. Having said this, R_s^2 is usually a good approximation for R^2 (especially in conditions of near-normal distributions). Kendall's τ, however, is not numerically similar to either r or r_s and so τ^2 does not tell us about the proportion of variance shared by two variables (or the ranks of those two variables).

Second, Kendall's τ is 66–75% smaller than both Spearman's r_s and Pearson's r, but r and r_s are generally similar sizes (Strahan, 1982). As such, if τ is used as an effect size it should be borne in mind that it is not comparable to r and r_s and should not be squared. A related issue is that the point-biserial and biserial correlations differ in size too (as we saw in this chapter, the biserial correlation was bigger than the point-biserial). In this instance you should be careful to decide whether your dichotomous variable has an underlying continuum, or whether it is a truly discrete variable. More generally, when using correlations as effect sizes you should remember (both when reporting your own analysis and when interpreting others) that the choice of correlation coefficient can make a substantial difference to the apparent size of the effect.



6.9. How to report correlation coefficents ①

Reporting correlation coefficients is pretty easy: you just have to say how big they are and what their significance value was (although the significance value isn't *that* important because the correlation coefficient is an effect size in its own right!). Five things to note are that: (1) if you follow the conventions of the American Psychological Association, there should be no zero before the decimal point for the correlation coefficient or the probability value (because neither can exceed 1); (2) coefficients are reported to 2 decimal places; (3) if you are quoting a one-tailed probability, you should say so; (4) each correlation coefficient is represented by a different letter (and some of them are Greek); and (5) there are standard criteria of probabilities that we use (.05, .01 and .001). Let's take a few examples from this chapter:

- ✓ There was a significant relationship between the number of adverts watched and the number of packets of sweets purchased, $r = .87$, p (one-tailed) $< .05$.
- ✓ Exam performance was significantly correlated with exam anxiety, $r = -.44$, and time spent revising, $r = .40$; the time spent revising was also correlated with exam anxiety, $r = -.71$ (all ps $< .001$).
- ✓ Creativity was significantly related to how well people did in the World's Biggest Liar competition, $r_s = -.37$, $p < .001$.
- ✓ Creativity was significantly related to how well people did in the World's Biggest Liar competition, $\tau = -.30$, $p < .001$. (Note that I've quoted Kendall's τ here.)
- ✓ The gender of the cat was significantly related to the time the cat spent away from home, $r_{pb} = .38$, $p < .01$.
- ✓ The gender of the cat was significantly related to the time the cat spent away from home, $r_b = .48$, $p < .01$.

Scientists, rightly or wrongly, tend to use several *standard* levels of statistical significance. Primarily, the most important criterion is that the significance value is less than .05; however, if the exact significance value is much lower then we can be much more confident about the strength of the effect. In these circumstances we like to make a big song and dance about the fact that our result isn't just significant at .05, but is significant at a much lower level as well (hooray!). The values we use are .05, .01, .001 and .0001. You are rarely going to be in the fortunate position of being able to report an effect that is significant at a level less than .0001!

When we have lots of correlations we sometimes put them into a table. For example, our exam anxiety correlations could be reported as in Table 6.3. Note that above the diagonal I have reported the correlation coefficients and used symbols to represent different levels of significance. Under the table there is a legend to tell readers what symbols represent. (Actually, none of the correlations were non-significant or had p bigger than .001, so most of these are here simply to give you a reference point – you would normally include symbols that you had actually used in the table in your legend.) Finally, in the lower part of the table I have reported the sample sizes. These are all the same (103), but sometimes when you have missing data it is useful to report the sample sizes in this way because different values of the correlation will be based on different sample sizes. For some more ideas on how to report correlations have a look at Labcoat Leni's Real Research 6.1.

Table 6.3 An example of reporting a table of correlations

	Exam Performance	*Exam Anxiety*	*Revision Time*
Exam Performance	1	−.44***	.40***
Exam Anxiety	103	1	−.71***
Revision Time	103	103	1

ns = not significant ($p > .05$), * $p < .05$, ** $p < .01$, *** $p < .001$

Labcoat Leni's Real Research 6.1

Why do you like your lecturers? ①

Chamorro-Premuzic, T., et al. (2008). *Personality and Individual Differences*, *44,* 965–976.

As students you probably have to rate your lecturers at the end of the course. There will be some lecturers you like and others that you hate. As a lecturer I find this process horribly depressing (although this has a lot to do with the fact that I tend focus on negative feedback and ignore the good stuff). There is some evidence that students tend to pick courses of lecturers whom they perceive to be enthusastic and good communicators. In a fascinating study, Tomas Chamorro-Premuzic and his colleagues (Chamorro-Premuzic, Furnham, Christopher, Garwood, & Martin, 2008) tested a slightly different hypothesis, which was that students tend to like lecturers who are like themselves. (This hypothesis will have the students on my course who like my lectures screaming in horror.)

First of all, the authors measured students' own personalities using a very well-established measure (the NEO-FFI) which gives rise to scores on five fundamental personality traits: Neuroticism, Extroversion, Openness to experience, Agreeableness and Conscientiousness. They also gave students a questionnaire that asked them to rate how much they wanted their lecturer to have each of a list of characteristics. For example, they would be given the description 'warm: friendly, warm, sociable, cheerful, affectionate, outgoing' and asked to rate how much they wanted to see this in a lecturer from −5 (they don't want this characteristic at all) through 0 (the characteristic is not important) to +5 (I really want this characteristic in my lecturer). The characteristics on the questionnaire all related to personality characteristics measured by the NEO-FFI. As such, the authors had a measure of how much a student had each of the five core personality characteristics, but also a measure of how much they wanted to see those same characteristics in their lecturer.

In doing so, Tomas and his colleagues could test whether, for instance, extroverted students want extrovert lecturers. The data from this study (well, for the variables that I've mentioned) are in the file **Chamorro-Premuzic.dat**. Run some Pearson correlations on these variables to see if students with certain personality characteristics want to see those characteristics in their lecturers. What conclusions can you draw?

Answers are in the additional material on the companion website (or look at Table 3 in the original article, which will also show you how to report a large number of correlations).

What have I discovered about statistics? ①

This chapter has looked at ways to study relationships between variables. We began by looking at how we might measure relationships statistically by developing what we already know about variance (from Chapter 1) to look at variance shared between variables. This shared variance is known as *covariance*. We then discovered that when data are parametric we can measure the strength of a relationship using Pearson's correlation coefficient, r. When data violate the assumptions of parametric tests we can use Spearman's r_s, or for small data sets Kendall's τ may be more accurate. We also saw that correlations can be calculated between two variables when one of those variables is a dichotomy (i.e., composed of two categories); when the categories have no underlying continuum then we use the point-biserial correlation, r_{pb}, but when the categories do have an underlying continuum we use the biserial correlation, r_b. Finally, we looked at the difference between *partial correlations*, in which the relationship between two variables is measured controlling for the effect that one or more variables has on both of those variables, and *semi-partial correlations*, in which the relationship between two variables is measured controlling for the effect that one or more variables has on only one of those variables. We also discovered that I had a guitar and, like my favourite record of the time, I was ready to 'Take on the World'. Well, Wales at any rate …

R packages used in this chapter

boot
ggm
ggplot2
Hmisc
Polycor
Rcmdr

R functions used in this chapter

boot()
boot.ci()
cor()
cor.test()
pcor()
pcor.test()
polyserial()
prop.table()
rcorr()
read.csv()
read.delim()
table()

Key terms that I've discovered

Biserial correlation
Bivariate correlation
Coefficient of determination
Correlation coefficient
Covariance
Cross-product deviations
Dichotomous
Kendall's tau
Partial correlation
Pearson correlation coefficient
Point-biserial correlation
Semi-partial correlation
Spearman's correlation coefficient
Standardization

Smart Alex's tasks ①

- **Task 1**: A student was interested in whether there was a positive relationship between the time spent doing an essay and the mark received. He got 45 of his friends and timed how long they spent writing an essay (**hours**) and the percentage they got in the essay (**essay**). He also translated these grades into their degree classifications (**grade**): in the UK, a student can get a first-class mark (the best), an upper-second-class mark, a lower second, a third, a pass or a fail (the worst). Using the data in the file **EssayMarks.dat** find out what the relationship was between the time spent doing an essay and the eventual mark in terms of percentage and degree class (draw a scatterplot too!). ①
- **Task 2**: Using the **ChickFlick.dat** data from Chapter 3, is there a relationship between gender and arousal? Using the same data, is there a relationship between the film watched and arousal? ①
- **Task 3**: As a statistics lecturer I am always interested in the factors that determine whether a student will do well on a statistics course. One potentially important factor is their previous expertise with mathematics. Imagine I took 25 students and looked

at their degree grades for my statistics course at the end of their first year at university: first, upper second, lower second or third class. I also asked these students what grade they got in their GCSE maths exams. In the UK, GCSEs are school exams taken at age 16 that are graded A, B, C, D, E or F (an A grade is better than all of the lower grades). The data for this study are in the file **grades.csv**. Carry out the appropriate analysis to see if GCSE maths grades correlate with first-year statistics grades. ①

Answers can be found on the companion website.

Further reading

Chen, P. Y., & Popovich, P. M. (2002). *Correlation: Parametric and nonparametric measures*. Thousand Oaks, CA: Sage.

Howell, D. C. (2006). *Statistical methods for psychology* (6th ed.). Belmont, CA: Duxbury. (Or you might prefer his *Fundamental Statistics for the Behavioral Sciences,* also in its 6th edition, 2007. Both are excellent texts that are a bit more technical than this book, so they are a useful next step.)

Miles, J. N. V., & Banyard, P. (2007). *Understanding and using statistics in psychology: A practical introduction*. London: Sage. (A fantastic and amusing introduction to statistical theory.)

Wright, D. B.,& London, K. (2009). *First steps in statistics* (2nd ed.). London: Sage. (This book is a very gentle introduction to statistical theory.)

Interesting real research

Chamorro-Premuzic, T., Furnham, A., Christopher, A. N., Garwood, J., & Martin, N. (2008). Birds of a feather: Students' preferences for lecturers' personalities as predicted by their own personality and learning approaches. *Personality and Individual Differences*, *44,* 965–976.