

Chapter 14: Association

S520

These notes are written to accompany Trosset chapter 14 (focusing mostly on sections 14.2.1 and 14.5.)

Bivariate normal random variables

Trosset section 14.2.

In bivariate data, each individual has two variables measured. These measurements might be of similar things (a first test and a second test) or of completely different things (income and height.) We want to study the **bivariate distribution** of such data. In general, if the two variables are related, it's not enough to describe both variables separately: we need a way of quantifying the relationship between the variables. We'll develop a way to do this in the context of **bivariate normal random variables**. These are pairs of random variables (X, Y) such that:

- X is normal;
- Y is normal;
- X and Y have a linear relationship (or no relationship.)

On a scatterplot, bivariate normal data looks like a circle or ellipse. (A more thorough check of bivariate normality would require QQ plots of each variable as well as the scatterplot.)

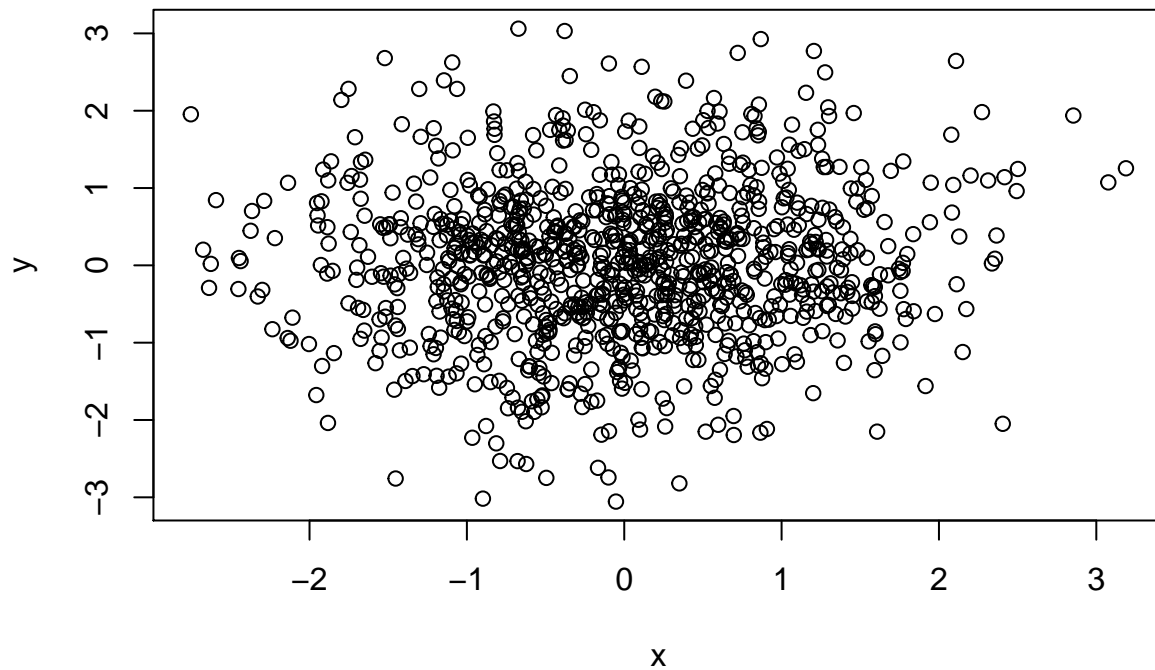
To study such objects, we'll use the functions written by Trosset that are in the `binorm.R` file on his webpage. To use these functions, we need to **source** them into R:

```
source("http://mypage.iu.edu/~mtrosset/StatInferR/binorm.R")
```

The `plot()` command in R draws a standard scatterplot when there are two input variables. Trosset's `binorm.scatter()` function draws a scatterplot of the data and adds an ellipse that gives the shape of the data cloud. (One slight complication is that the two variables must be a matrix; we use the `cbind()` ("column bind") function to create one.) We'll simulate some bivariate normal data and apply this function. We'll also calculate a numerical measure of association called the **correlation**, leaving its definition for later.

Firstly, no association:

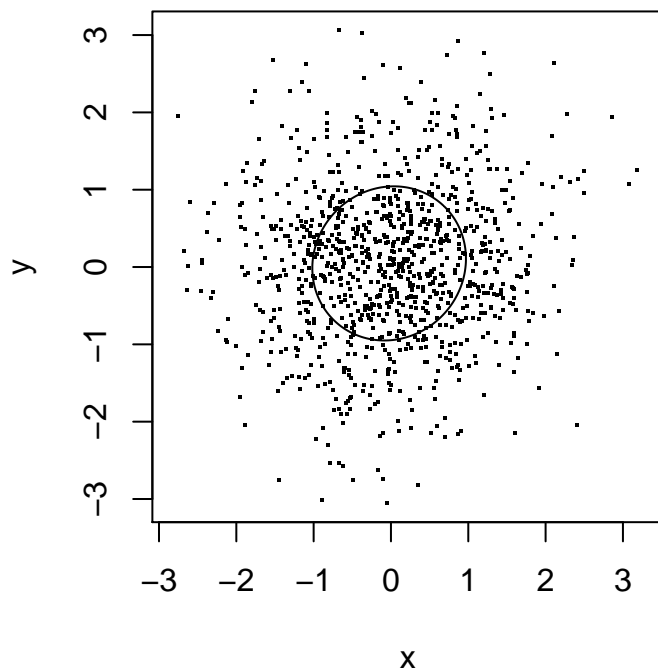
```
x = rnorm(1000)
y = rnorm(1000)
plot(x, y)
```



The standard scatterplot is a bit of a mess. Using `binorm.scatter()` instead:

```
binorm.scatter(cbind(x, y))
```

Scatter Diagram



```
cor(x, y)
```

```
## [1] 0.06137515
```

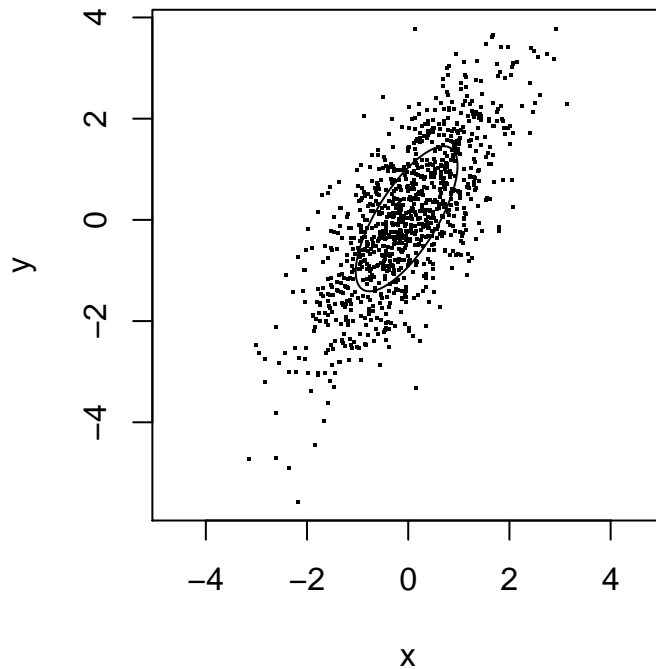
Note that the ellipse is not designed to contain all of the data points – instead, it aims to indicate the shape of the data cloud. In this case, the ellipse is basically a circle. There's no obvious association between the

two variables (and there shouldn't be, as we created them independently.)

Now we'll simulate an example where there *is* a relationship between x and y .

```
# Strong positive linear association
x = rnorm(1000)
y = x + rnorm(1000)
binorm.scatter(cbind(x, y))
```

Scatter Diagram



```
cor(x, y)
```

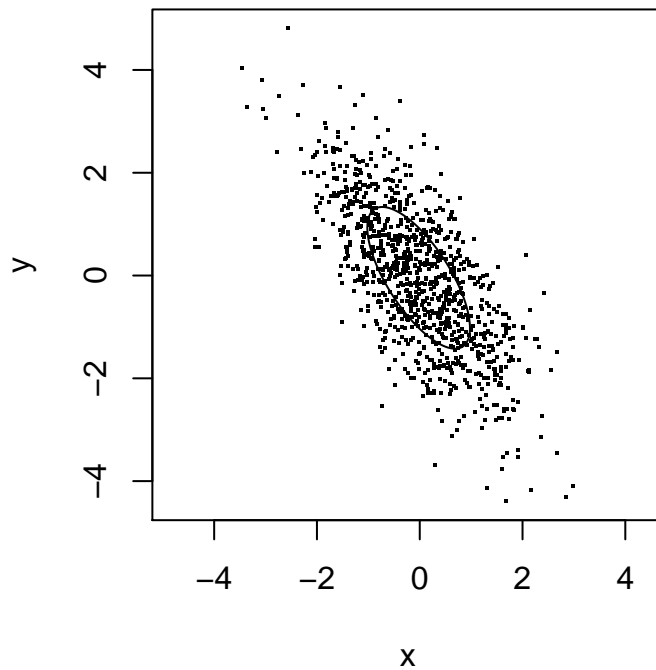
```
## [1] 0.7297232
```

The relationship is that y is just x with some random noise added. Without the random noise, the relationship would just be $y = x$, which is a *linear* association. The association is *positive*: as x goes up, y tends to go up as well. Finally, the association is quite *strong*: the points are relatively close to the line, or to put it another way, it's relatively clear where where to draw the line on this graph.

Now we'll draw a graph that points the other way.

```
x = rnorm(1000)
y = -x + rnorm(1000)
binorm.scatter(cbind(x, y))
```

Scatter Diagram



```
cor(x, y)
```

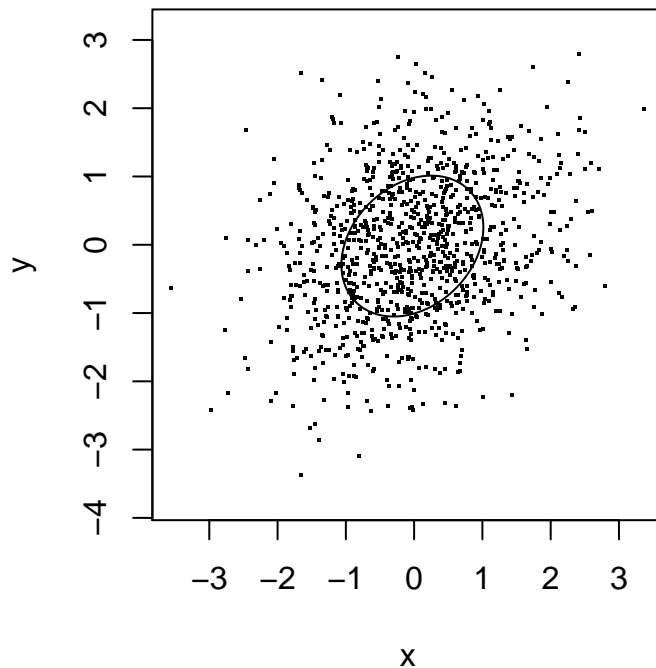
```
## [1] -0.7083118
```

The association is still linear and strong, but this time the line points downward, so the association is *negative*.

Here's an example where the association is weaker:

```
x = rnorm(1000)
y = x/4 + rnorm(1000)
binorm.scatter(cbind(x, y))
```

Scatter Diagram



```
cor(x, y)
```

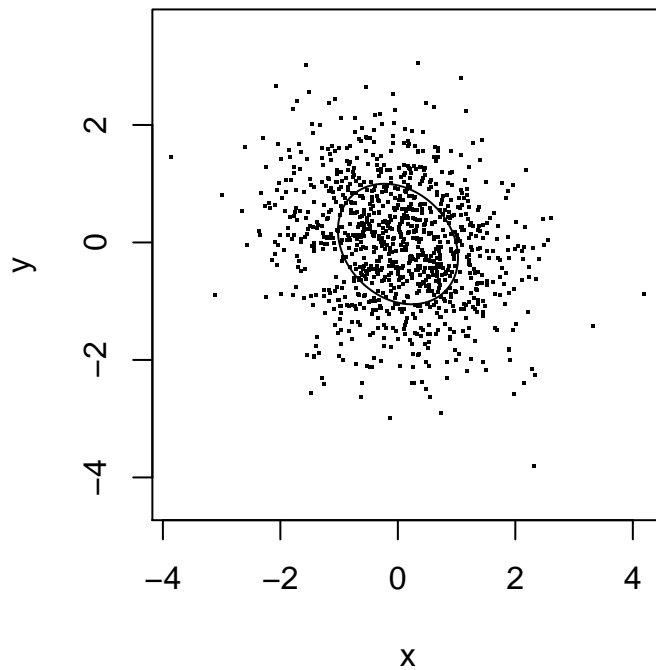
```
## [1] 0.2660259
```

The association is harder to see here, but looking carefully, there are quite a few points in the bottom left and top right, and hardly any points in the top left and bottom right. So if we drew a line to summarize the data, it would go from bottom left to top right, i.e. its slope is positive. Furthermore, there's no obvious reason to draw a curve rather than a line. So there's still a positive linear association, but it's fairly weak.

Here's one that points the other way:

```
x = rnorm(1000)
y = -x/4 + rnorm(1000)
binorm.scatter(cbind(x, y))
```

Scatter Diagram



```
cor(x, y)
```

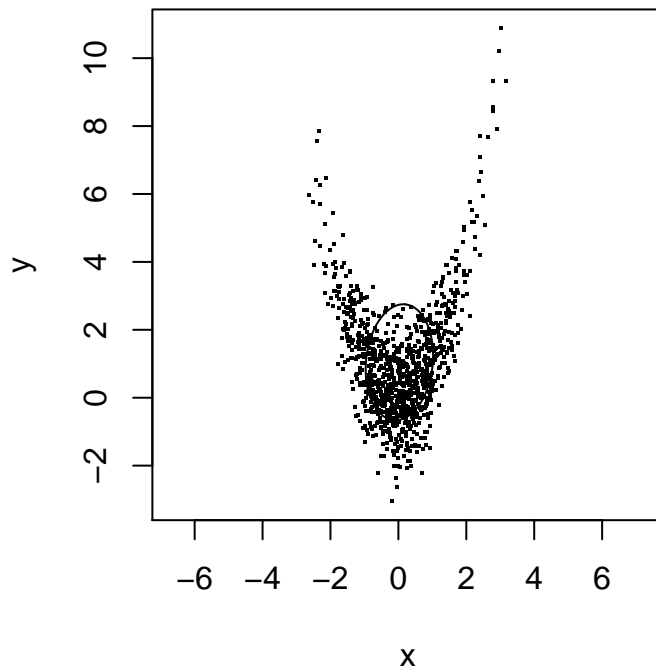
```
## [1] -0.2293685
```

There are more top left/bottom right points than bottom left/top right points. There's a weak negative linear association.

Finally, here's one that **isn't** bivariate normal.

```
x = rnorm(1000)
y = rnorm(1000) + x^2
binorm.scatter(cbind(x, y))
```

Scatter Diagram



The relationship is quadratic rather than linear, violating a condition for the bivariate normal. We see the ellipse does a horrible job of describing the shape of the data.

Real data: Anorexia treatments

Let's look at the anorexia data (first introduced in Trosset ch. 12.5.) The data is on Trosset's webpage and is also in a slightly more user-friendly format in the file `anorexia.txt`. The data is from an experiment with three treatments for anorexia: Cognitive, Family, and Standard. There are two numerical measurements on each patient: weight before the treatment and weight after the treatment (both in pounds.) Here, we'll just pick out the individuals who got the standard treatment.

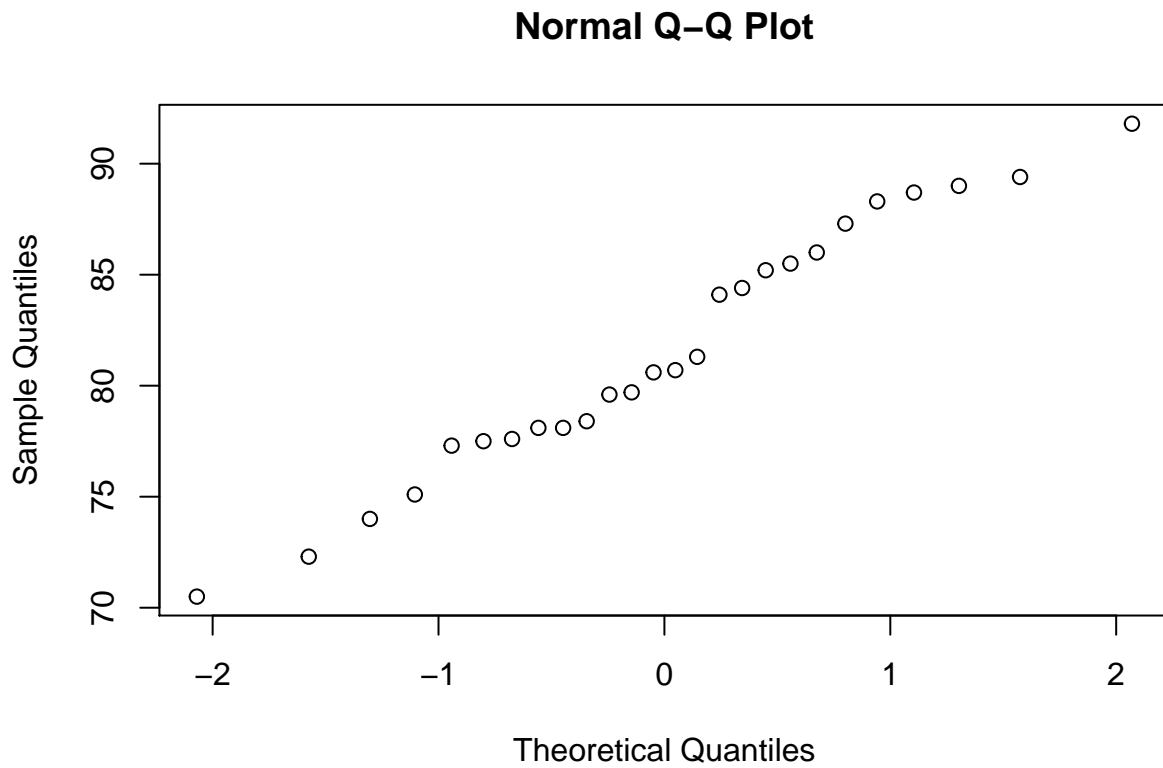
```
anorexia = read.table("anorexia.txt", header = TRUE)
Standard = anorexia[anorexia$Treatment == "Standard", ]
```

We create three variables called `Before`, `After`, and `Diffs`, where the latter is the change in weight (after minus before.)

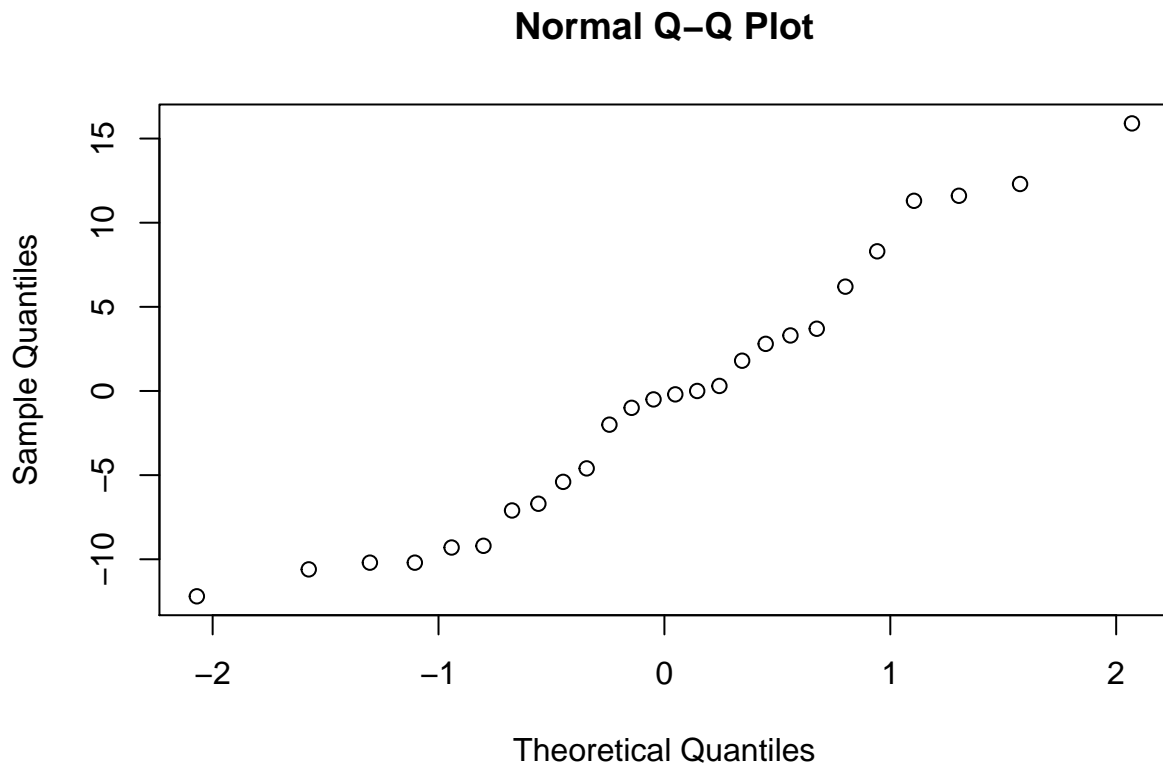
```
Before = Standard$Before
After = Standard$After
Diffs = After - Before
```

Let's consider `Before` and `Diffs`. Together, are these two variables well-approximated by a bivariate normal distribution? We can first check normal QQ plots:

```
qqnorm(Before)
```



```
qqnorm(Diffs)
```

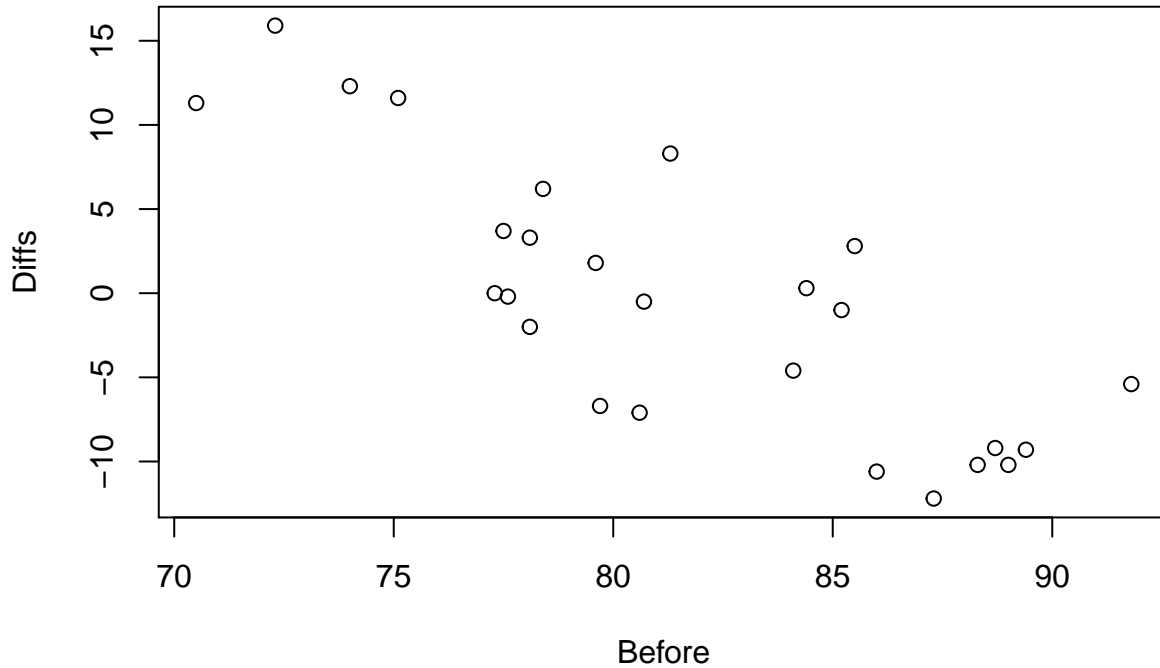


As always with small sample sizes, it's hard to tell if the data is really close to normal or not. But at least

here we see no extreme skewness or outliers that would contradict approximate normality.

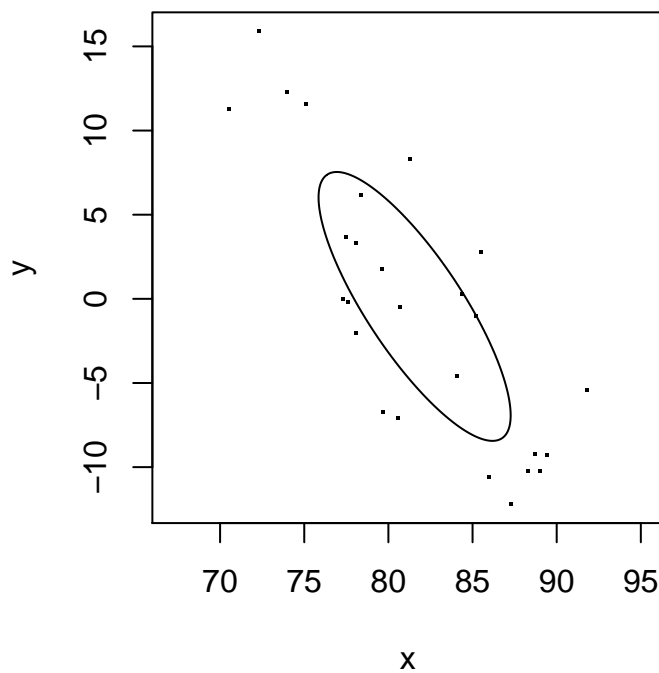
Now we need to draw a scatterplot. Whether you use the built-in `plot()` function or `binorm.scatter()` is personal preference.

```
plot(Before, Diffs)
```



```
binorm.scatter(cbind(Before, Diffs))
```

Scatter Diagram



Perhaps surprisingly, patients who were lighter than the average before the treatment tended to gain weight,

while patients who were heavier than the average before the treatment tended to lose weight. There seems to be a strong negative linear relationship between the patient's weight before treatment and their change in weight. The ellipse gives a pretty good description of the shape of the data. The bivariate normal seems like a reasonable approximation for this data.

Correlation

Here's **Pearson's product-moment correlation coefficient**, generally referred to as simply “the correlation” and denoted $\hat{\rho}$ or r :

```
cor(Before, Diffs)
```

```
## [1] -0.8102515
```

What does this number mean? The formula is on Trosset p. 356:

$$r = \frac{1}{n-1} \sum_{i=1}^n \left[\left(\frac{x_i - \bar{x}}{s_x} \right) \left(\frac{y_i - \bar{y}}{s_y} \right) \right]$$

So firstly, we change the x_i observation into “standard units” by subtracting the sample mean of x , then dividing by its sample standard deviation:

```
mean.before = mean(Before)
sd.before = sd(Before)
z.before = (Before - mean.before)/sd.before
```

We then similarly change the y_i observations to standard units:

```
mean.diff = mean(Diffs)
sd.diff = sd(Diffs)
z.diff = (Diffs - mean.diff)/sd.diff
```

We then multiply the standardized x 's by the standardized y 's, and take the average of these products. The only catch is that when taking the “average”, we sum up and divide by $n-1$ instead of n , since $n-1$ is in the denominator of the sample SD formula.

```
n = length(Before)
sum(z.before * z.diff)/(n - 1)
```

```
## [1] -0.8102515
```

We see this gives the same numbers as the `cor()` function. While it is character-building to find the correlation the long way, realistically you'll use `cor()`.

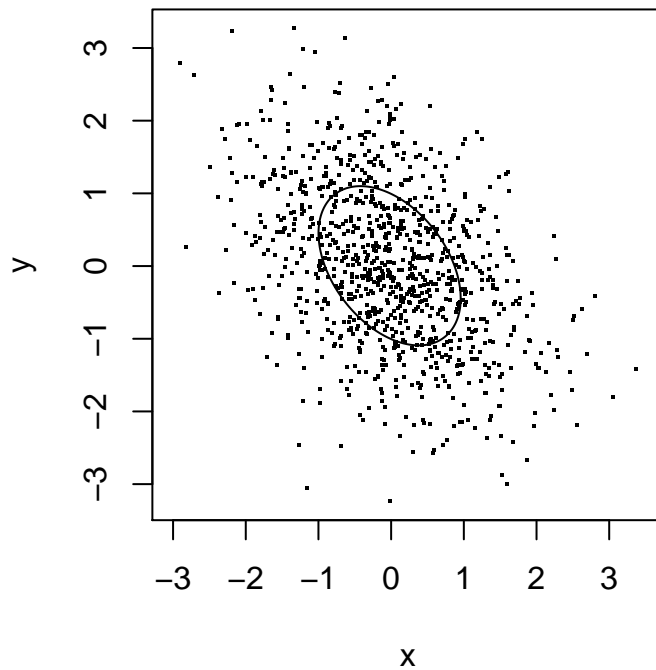
It can be shown that the correlation formula will always produce a number between -1 and $+1$. But what does this correlation tell us?

The correlation is a measure of *linear* association. The sign (positive or negative) tells us whether the line points up or down. If the linear association is strong, the correlation will be closer to $+1$ or -1 . If the linear association is weak, the correlation will be closer to 0. (Exactly 0 means no correlation at all, though this is rare in real data – you'll almost always get some sample correlation, even if the variables are truly independent, just by luck.)

When the data actually is bivariate normal, correlation does a good job of measuring the strength of the association:

```
x = rnorm(1000)
y = -0.5 * x + rnorm(1000)
binorm.scatter(cbind(x, y))
```

Scatter Diagram



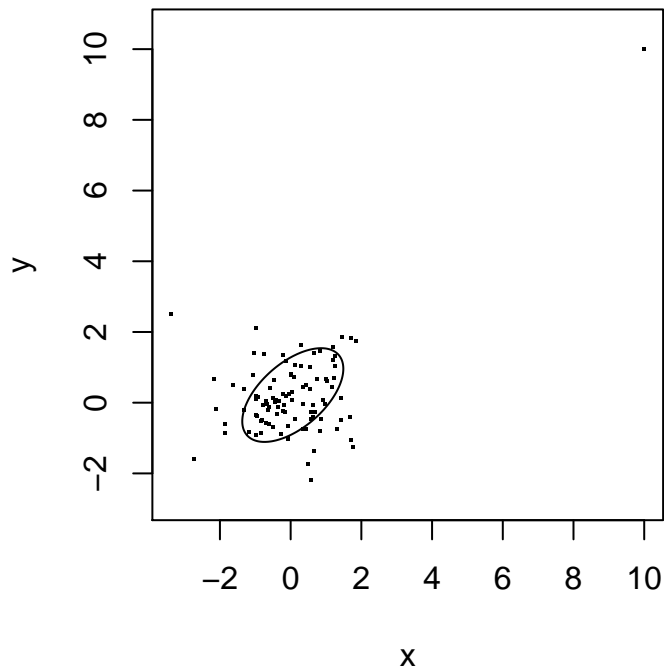
```
cor(x, y)
```

```
## [1] -0.41017
```

We have to be a bit more cautious when the data is skewed or heavy-tailed, because like so many other statistics, correlation is sensitive to outliers.

```
x = c(rnorm(99), 10)
y = c(rnorm(99), 10)
binorm.scatter(cbind(x, y))
```

Scatter Diagram



```
cor(x, y)
```

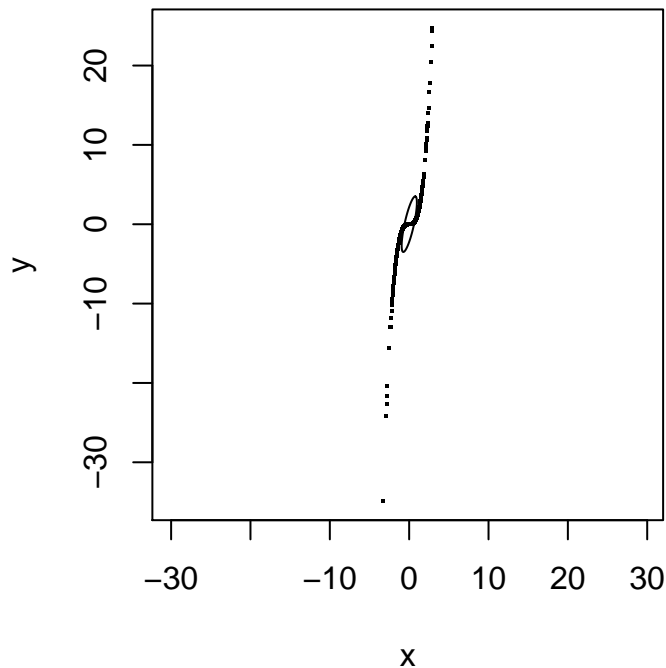
```
## [1] 0.5605533
```

The correlation in the graph above is moderately strong, but it's entirely due to one observation at (10, 10). Check your scatterplot for outliers before interpreting the correlation.

When the relationship between two variables is nonlinear, the standard Pearson's correlation can be misleading:

```
x = rnorm(1000)
y = x^3
binorm.scatter(cbind(x, y))
```

Scatter Diagram



```
cor(x, y)
```

```
## [1] 0.7780356
```

The correlation is quite high, indicating a strong linear association. However, the association is actually *perfect*: the points lie exactly on the curve $y = x^3$.

An alternative to Pearson's correlation that works well for data that's nonlinear but is still **monotonic** (i.e. always rising or always falling) is **Kendall's tau**:

```
cor(x, y, method = "kendall")
```

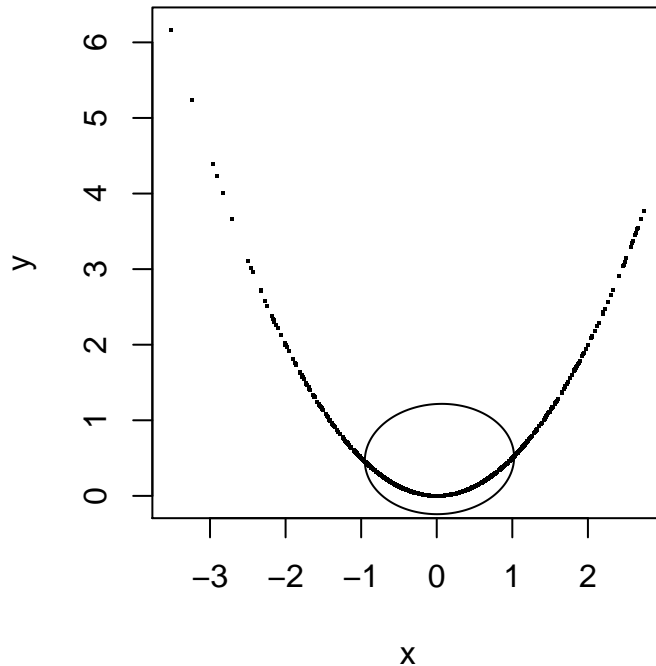
```
## [1] 1
```

We won't focus on Kendall's tau here other than noting that it exists. See Trosset 14.3 if you're interested in the details.

Finally, if the relationship is non-monotonic (e.g. U-shaped), neither Pearson's correlation nor Kendall's tau do a good job of describing the strength of the association.

```
x = rnorm(1000)
y = 0.5 * x^2
binorm.scatter(cbind(x, y))
```

Scatter Diagram



```
cor(x, y)
```

```
## [1] 0.03195488
```

```
cor(x, y, method = "kendall")
```

```
## [1] 0.04113714
```

The data cloud isn't an ellipse and the trend isn't a straight line. Neither Pearson's correlation nor Kendall's tau picks up the perfect quadratic relationship.

Correlation is not causation

You've heard this before, I'm sure. Well, it's true. For trivial counterexamples, see: <http://tylervigen.com/old-version.html>

By *causation*, we usually want to know if changes in a variable x cause changes in another variable y . However, if two variables x and y are associated, any of the following could be the case:

- Changing x causes changes in y ;
- Changing y causes changes in x ;
- x and y both cause changes in each other;
- x and y don't have any direct causal relationship, but changes in z cause changes in both x and y , so x and y end up correlated anyway;
- The association is just luck.

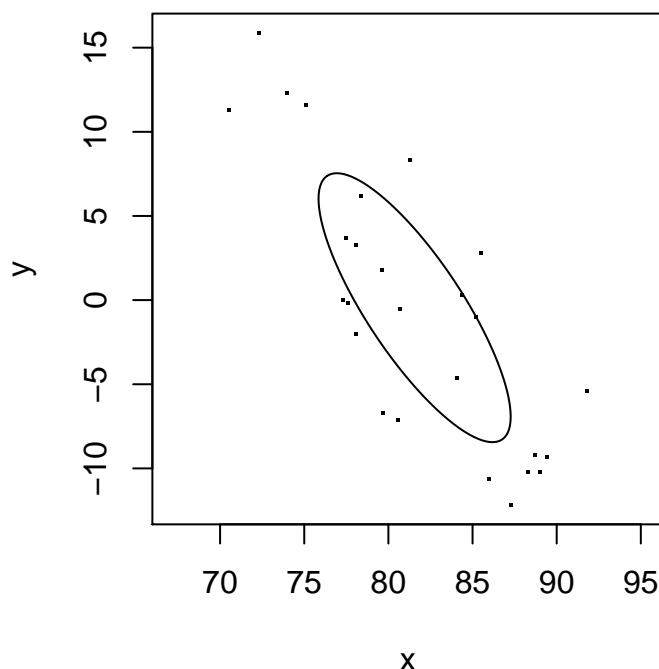
Or it could be some combination of the above. Correlation alone doesn't let you distinguish between these options. To make a statement about cause-and-effect, do a randomized controlled experiment, or if that's not ethical, do a study that looks as much like a randomized controlled experiment as possible (set up your treatment and control groups prospectively, minimize differences between the groups besides the treatment, and observe changes over time.)

Anorexia treatments revisited

Return to the scatterplot of weights before and weight changes for patients who got the standard treatment:

```
binorm.scatter(cbind(Before, Diffs))
```

Scatter Diagram



```
cor(Before, Diffs)
```

```
## [1] -0.8102515
```

Lighter patients tended to gain weight, while heavier patients tended to lose weight. The association seems very strong: the correlation is -0.81 . It's tempting to give a causal explanation – maybe the treatment worked for the lighter patients but backfired for the heavier patients. But do we necessarily need such a causal explanation?

To investigate further, we examine the relationship between weights before and weights after the standard treatment.

```
summary(Before)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  70.50   77.72   80.65   81.56   85.88   91.80
```

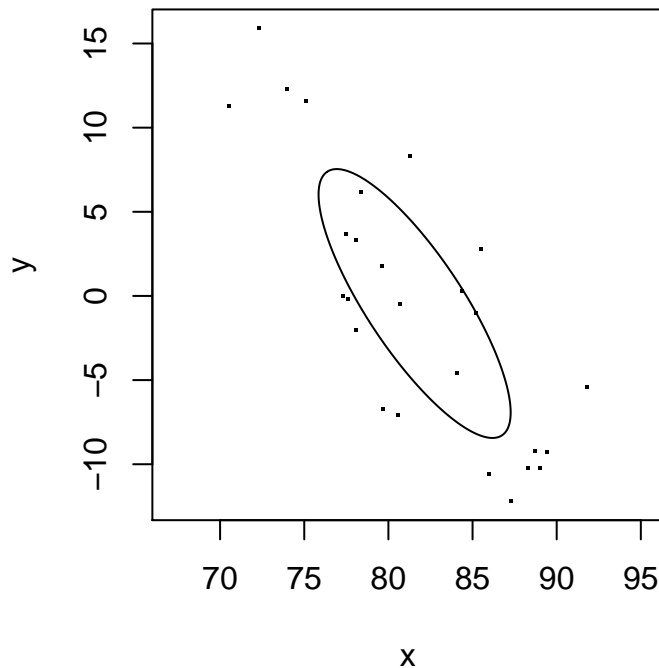
```
summary(After)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  73.00   77.58   80.70   81.11   84.67   89.60
```

We first note that the sample means and medians are very close.

```
binorm.scatter(cbind(Before, Diffs))
```

Scatter Diagram



```
cor(Before, After)
```

```
## [1] -0.1614161
```

The correlation here is much weaker. We haven't focused on significance tests for correlation because the author thinks bivariate tests are more clearly introduced in the context of linear regression (ch. 15.) However, on the rare occasions you do have approximately bivariate normal data, the function `cor.test()` can test the null hypothesis that the correlation is zero:

```
cor.test(Before, After)
```

```
##
## Pearson's product-moment correlation
##
## data: Before and After
## t = -0.80128, df = 24, p-value = 0.4308
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
## -0.5164755 0.2410046
## sample estimates:
## cor
## -0.1614161
```

The not-small P -value indicates the data is consistent with no (population) correlation at all between weights before and weights after the standard treatment. Even if there is a population correlation, we can't be sure from the data whether it's positive or negative.

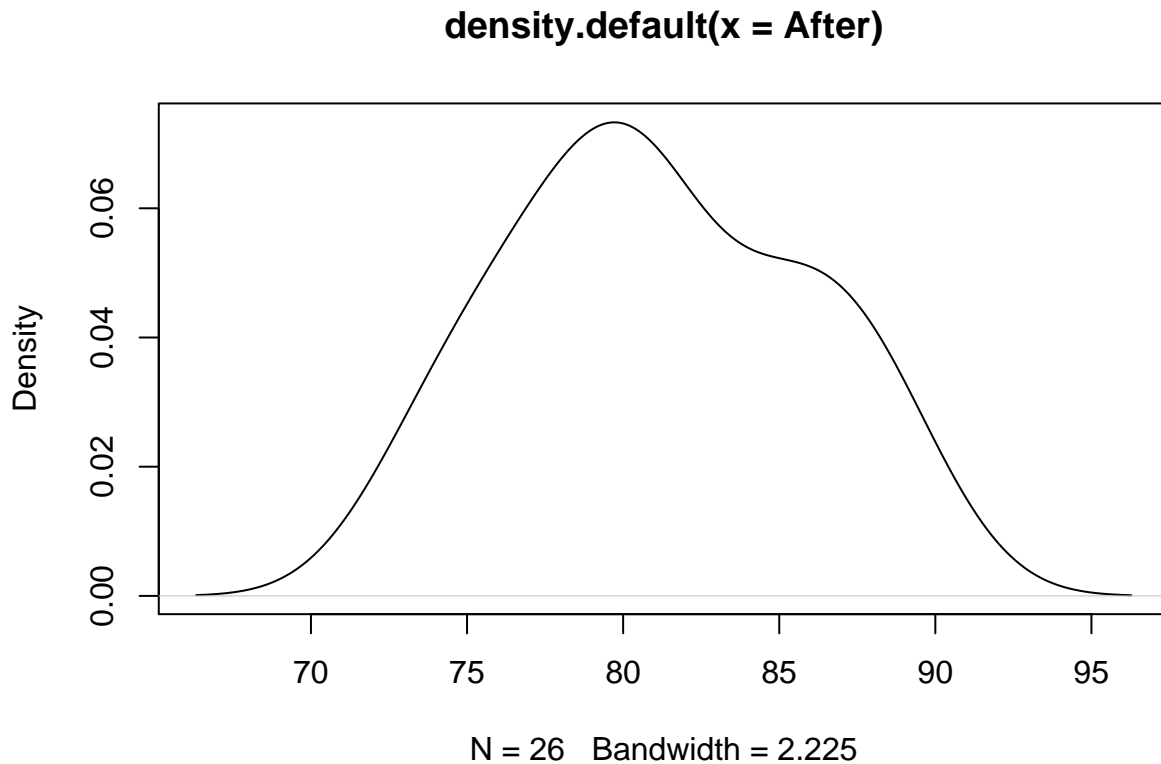
We've made the following observations:

1. The relationship between "weights before" and "weights after" is weak or negligible.
2. The relationship between "weights before" and "change in weights" is negative.

But if you think about it carefully, the second point follows from the first. If weight before and weights after

are basically unrelated, then it doesn't really matter what your weight was before the treatment. So if we were making a prediction for weight after, we'd make essentially the same prediction regardless of your weight before. What should that prediction be?

```
plot(density(After))
```



As with most distributions, the PDF of “weight after” is concentrated around its mean. So that means a good prediction for weight after is that variable’s mean: about 81 pounds.

This implies: - If your weight before was less than 81 pounds, we predict your weight after will be about 81 pounds. - If your weight before was more than 81 pounds, we predict your weight after will be about 81 pounds.

In other words, if you were lighter than average before, we predict a *positive* change in weight to bring you closer to the mean. If you were heavier than average before, we predict a *negative* change in weight to bring you closer to the mean. This is an example of a very general phenomenon called **regression toward the mean** or just **regression**, which, not coincidentally, is one of the words in the title of the next chapter.