Minilab 5a Worksheet

Linear Regression: The Intuitive Idea

We have seen that a *scatterplot* is useful for visualising the relationship between two quantitative variables. Sometimes we wish to predict the value of one of the variables using only the value of the other variable. One way to do this is to use a "line of best fit" to model the data.

In this minilab, we explore the **intuitive idea** behind linear regression, i.e., fitting the best possible straight line through a scatterplot relating two quantitative variables. This is one type of *statistical model*.

Image from http://www.game-changer.net/2018/12/04/3-conditions-necessary-to-trust-your-intuition/

1. What is Linear Regression?

"All models are wrong, but some are useful" — George Box

Suppose we wish to predict the value of a single *response* (output) variable Y based on a single *predictor* (input) variable X. The aim is to establish a relationship (a mathematical formula) between the predictor variable and the response variable so that we can use this formula to estimate the value of the response when only the value of the predictor is known.

In *linear regression*, we want to find the straight line through the *observed* data points $\{(x_i, y_i)\}$ that "best fits" the data.

The mathematical equation for a line can be written as

$$y = a + bx$$

where a is the *intercept* and b is the *slope*.

If we are given the values of a and b then for any value of x we can *predict* the corresponding value of y, i.e., for each x_i the *predicted* value is (called "y hat")

$$\hat{y}_i = a + bx_i$$

We are then concerned the question:

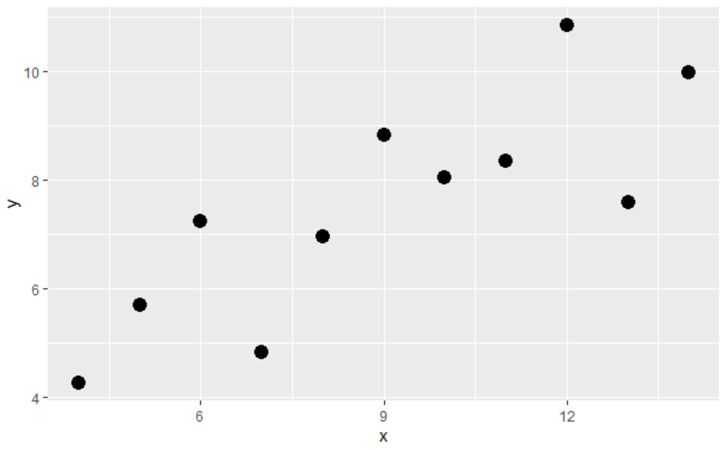
How close is each predicted value is to its actual (or observed) value?

That is, how close is each $\hat{y}_i$ value is to its corresponding y_i value. This is called a *residual* (or *error)* and is calculated as

$$e_i = y_i - \hat{y}_i$$

(1) Let us have a look again at the first dataset from Anscombe's quartet (the data is built into R) and plot the corresponding scatterplot using ggplot.

```
library(tidyverse)
# Anscombe's quartet (first dataset)
x = anscombe[,1]
y = anscombe[,5]
ggplot(NULL,aes(x=x,y=y)) +
  geom_point(size=4)
```



(2) If we were to “guesstimate” a *slope* for a straight line through this dataset, the points range from 4 to 14 horizontally and from approximately 4 to 11 vertically, so we would expect a slope of approximately

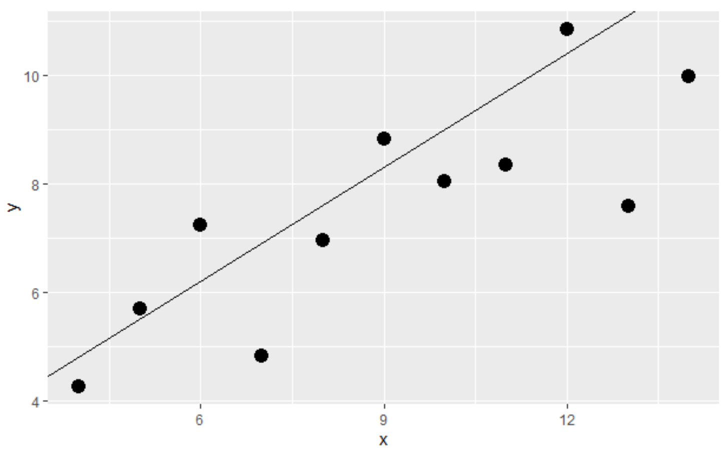
$$b = \frac{\text{rise}}{\text{run}} = \frac{11-4}{14-4} = \frac{7}{10} = 0.7$$

We might also guesstimate a value for the *intercept* of around $a = 2.0$ (perhaps you can imagine the plot axes to extend further to the left and extrapolate a straight line line until it hits the y-axis).

So, one possible approximation to this dataset is the line: $y = 2.0 + 0.7x$

(3) Given $a = 2.0$ and $b = 0.7$, we can calculate the corresponding residuals (observed minus predicted values of y) as follows.
Notice (in this case) that some of the residuals are positive and some are negative, which (reassuringly) indicates that the proposed line $y = 2 + 0.7x$ does pass “through” the cloud of data points (some points are above the line and some below).

```
a = 2.0
b = 0.7
residuals = y-(a+b*x)
residuals
ggplot(NULL,aes(x=x,y=y)) +
  geom_point(size=4) +
  geom_abline(slope=b,intercept=a)
```



2. Which line is the line of "best fit"?

If we change the value of a or b we change the line that is supposed to be representing the data. So the key question is:

How can we find the values of a and b that give the line of "best fit"?

Statisticians define what is meant by "best" as the values of a and b that **minimise** the sum of the squares of the residuals, i.e., make

$$\sum e_i^2 = e_1^2 + e_2^2 + \cdots + e_n^2$$

as **small** as possible. This penalises points that are (vertically) far from the line a lot and penalises points that are (vertically) close to the line very little.

(1) We now want to attempt to do this **visually** using the following R code.

```
b = 0.7
a = mean(y)-b*mean(x)
residuals = y-(a+b*x)
ggplot(NULL,aes(x=x,y=y)) +
  geom_point(size=4) +
  geom_abline(intercept=a,slope=b) +
  geom_rect(aes(xmin=x,xmax=x-residuals,
                ymin=y,ymax=y-residuals),
                fill="green",alpha=0.5) +
  coord_fixed() +
  xlim(3,17) +
  ylim(3,12)
SSR = sum(residuals^2)
SSR
```

In this R code, the `coord_fixed()` ensures that the two axes have the same scale, e.g., the distance on screen between 5 and 10 is the same on the horizontal and vertical axes. *So squares do appear as squares.* The `xlim` and `ylim` just fix the x-axis and y-axis limits so that in the further plots we make below we see the same axes each time.

In the code, we propose a value for b and then calculate a corresponding value for a so that the line goes through the point $(\bar{x}, \bar{y})$ which seems reasonable.

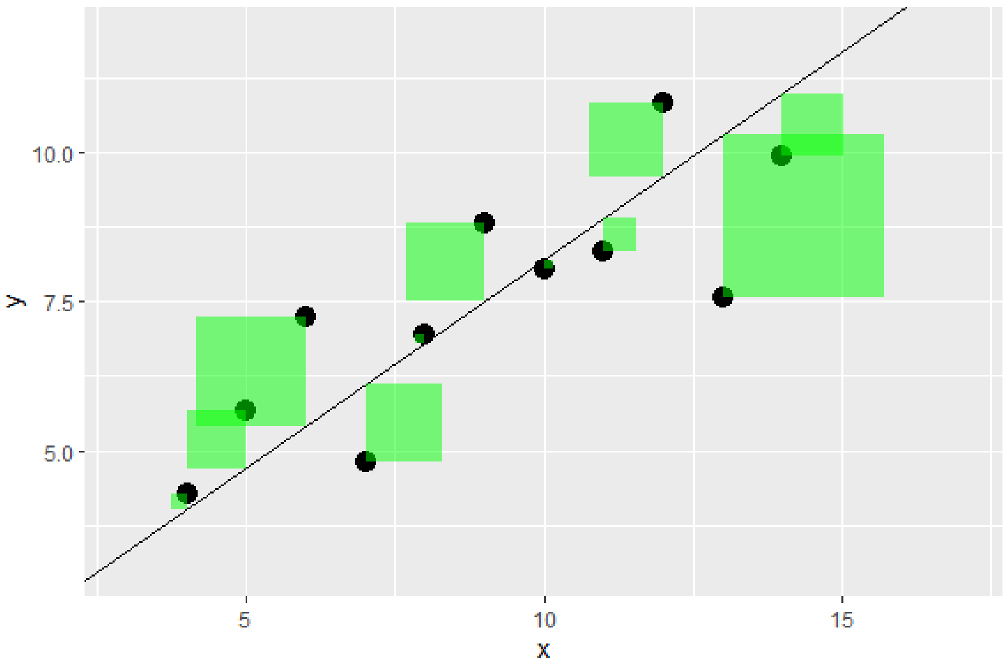
Each data point (x_i, y_i) is represented by a black dot. Each black dot is connected by a *vertical* line segment (the vertical side of the green square) to the corresponding predicted point on the line, i.e., $(x_i, \hat{y}_i)$.

The height and width of each green square is the value of the residual $e_i = y_i - \hat{y}_i$ so the

area of each green box

is the

square of the residual (i.e., e_i^2).

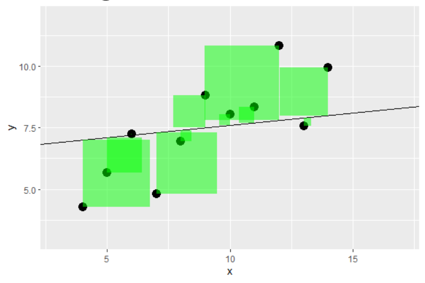
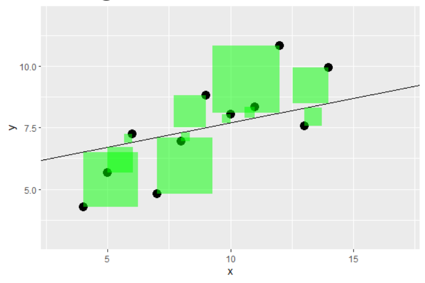
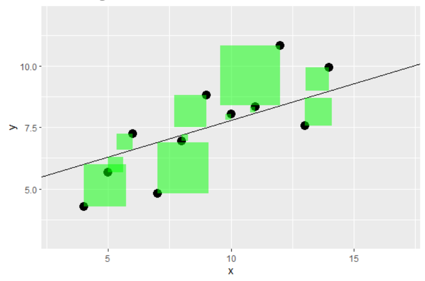
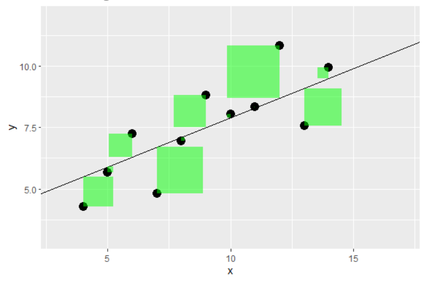


We want the

<u>sum</u> of the areas

of the green squares
to be as **small as possible**.

(2) Now suppose we experiment with changing the value of b in the R code above. The plots below show that as b increases from $b = 0.1$ to $b = 0.4$, the sum of squared residuals (SSR) decreases. The scales are the same on all these plots, so we want the value of b that gives the smallest amount of green area.

$b = 0.1$ gives $SSR = 31.37$ (2dp) 	$b = 0.2$ gives $SSR = 23.69$ (2dp) 
$b = 0.3$ gives $SSR = 18.17$ (2dp) 	$b = 0.4$ gives $SSR = 14.86$ (2dp) 

Exercise. Continue to <u>increase</u> the value of b in the R code above, and note what happens to the SSR value. What value of b appears to give the **<u>smallest</u>** SSR value? *You might want to try wrapping the R code in a loop (for b) to automate the process.*

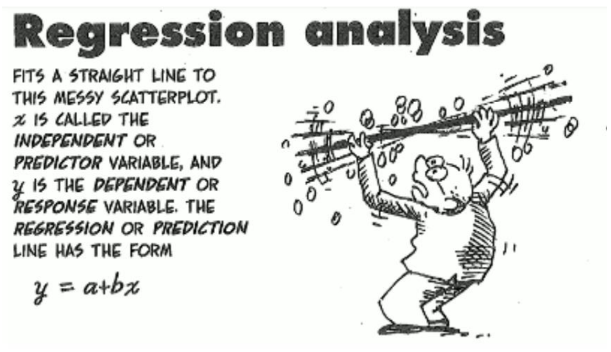

Regression analysis
FITS A STRAIGHT LINE TO THIS MESSY SCATTERPLOT. x IS CALLED THE INDEPENDENT OR PREDICTOR VARIABLE, AND y IS THE DEPENDENT OR RESPONSE VARIABLE. THE REGRESSION OR PREDICTION LINE HAS THE FORM
y = a+bx

Image from: https://medium.com/analytics-vidhya/linear-regression-simple-single-multiple-fb8a1a678168

Exercise. A dataset on using wetland systems to treat wastewater uses biochemical oxygen demand (BOD) mass *loading* (the x variable) to predict BOD mass *removal* (the y variable).

```
library(tidyverse)
loading = c(3,8,10,11,13,16,27,30,35,37,38,44,103,142)
removal = c(4,7, 8, 8,10,11,16,26,21, 9,31,30, 75, 90)
wetland = tibble(loading,removal)
x = loading
y = removal
```

(a) Use ggplot to construct a scatterplot of this dataset. Roughly estimate the slope of the line of best fit.

(b) Modify the R code on page 4 to refine the estimate of b, starting from $b = 1$. Make sure you modify the xlim() and ylim() so that you can see all the green squares. What value of b appears to give the **smallest** SSR value? *You might want to try wrapping the R code in a loop to automate the process.*

3. Is there a formula to calculate the best values of a and b?

Formulas for the values of a and b that minimise the sum of squared residuals can be found using calculus (we are not expecting you to know any calculus in this module), giving:

$$b = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2}$$

$$a = \bar{y} - b\bar{x}$$

In R, we can therefore calculate a and b directly using these formulas.

```
x = anscombe[,1]
y = anscombe[,5]
b = sum((x-mean(x))*(y-mean(y)))/sum((x-mean(x))^2)
a = mean(y)-b*mean(x)
b
a
```

Conclusion. The best-fit values of a and b are: $a = 3.000091$ and $b = 0.5000909$.

Exercise. For the wetland dataset, using mass *loading* (the x variable) to predict BOD mass *removal* (the y variable), calculate the best-fit values of a and b and write down the line of best fit (in terms of *removal* and *loading*).

Exercise. The "cars" dataset is also built in to R. It gives the *speed* (mph) of cars and the corresponding distances (*dist*) taken to stop (feet). These are all cars from the 1920s (see https://rdrr.io/r/datasets/cars.html).

Image from: https://www.supercars.net/blog/cars-of-the-1920s/

```
library(tidyverse)
summary(cars)
x = cars$speed
y = cars$dist
```

(a) Plot a scatterplot of stopping distance vs speed (this means that *dist* is on the vertical axis and *speed* is on the horizontal axis). Add a title to your plot and label the horizontal and vertical axes. Roughly estimate the slope of the line of best fit.

(b) Calculate the *correlation coefficient* between stopping distance and speed. How would you describe the relationship between the speed and stopping distance?

(c) Suppose we wish to use *speed* (the x variable) to predict stopping *dist* (the y variable). Calculate the best-fit values of a and b and write down the line of best fit (in terms of *dist* and *speed*). For these 1920s cars, what is the approximate increase in stopping distance required for a 1mph increase in speed?

Summary

In this minilab, we have developed some **intuitive understanding** of linear regression, i.e., finding the "line of best fit". We define the idea of "best fit" carefully in terms of minimising the sum of the squared residuals. Changing the value of b changes the *slope* of the line through the point $(\bar{x}, \bar{y})$ and we produced an intuitive visualisation of the area of squares to see how the sum of squared residuals depends on the line being fitted. Finally, we used a mathematical formula to calculate the best-fit coefficients.

Important! —

This minilab helps us see what calculations are performed in order to find the line-of-best-fit but we will NEVER do these calculations ourselves after this minilab.

In practice, R takes care of all the calculations for us, and we can concentrate on interpretation of results. We will see this in the next few minilabs.