Multiple Regression

Bob O'Hara

Contents

This week: Multiple Regression

We will look at

- explaining our dependent variable with more than one explanatory variable
- how to fit these models in R
- what a design matrix is (this will be helpful later)
- why and how to fit a polynomial model

More Monsters

Begin with the intro video

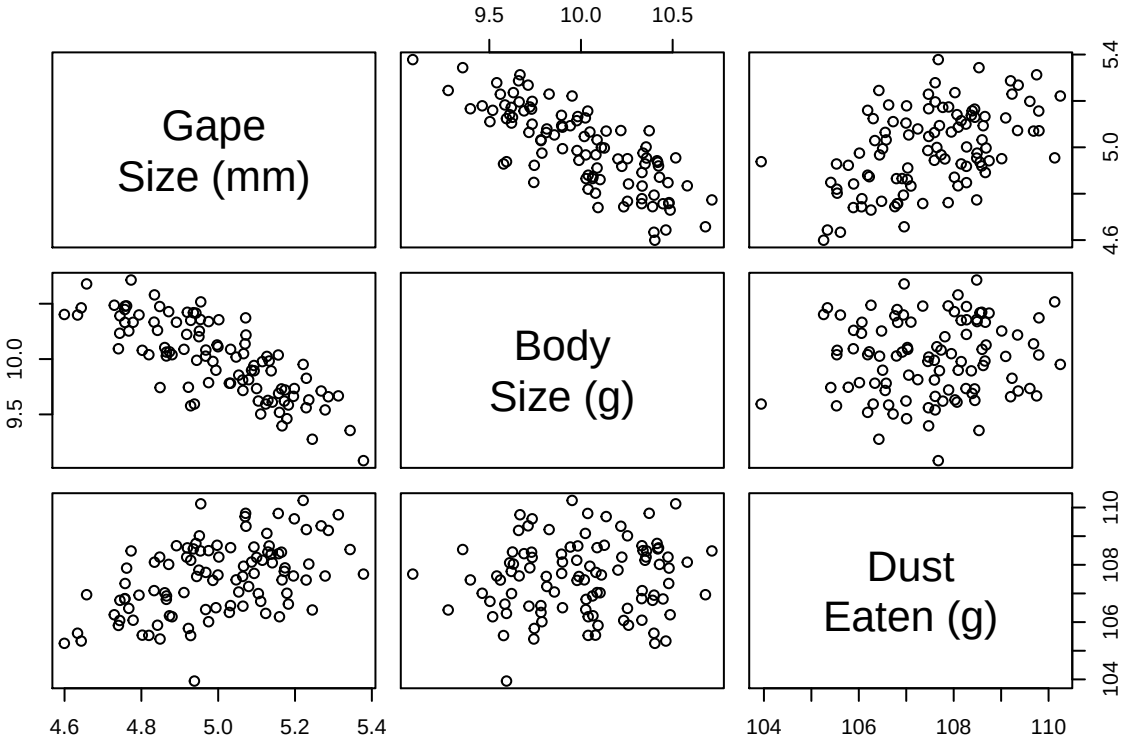
Click here or watch below

In the cellar of the museum in Frankfurt we had a population of Schey. These are small creatures that lurk in the dark and eat ancient dust and stale cobwebs. Some of us wanted to know more about them, and whether they could be trained to clean the museum collections.

We caught 100 and measured the amount of dust they could eat in 5 mins, and wanted to explain that by their body size, their gape size (i.e. how large their mouths are).

The Data

```
File <- "https://www.math.ntnu.no/emner/ST2304/2021v/Week07/ScheyData.csv"
Schey <- read.csv(File)
plot(Schey, labels=c("Gape\nSize (mm)", "Body\nSize (g)", "Dust\nEaten (g)"))
```



Before we have looked at regression against a single covariate. Now we want to look at regression against two covariates: the extension to more than 2 is straightforward.

This is a more common situaiton in practice than regression against a single covariate. Several factors often affect a response, so we need to take this into account. Even worse, they might interact, i.e. the effect of one covariate might depend on the value of another. Sometimes we are interested in all of the effects, at other times we are only interested in some, but we are worried that there are others that might have an effect, and need to be included in the analysis (not including them can bias the results, and also make the estimates worse).

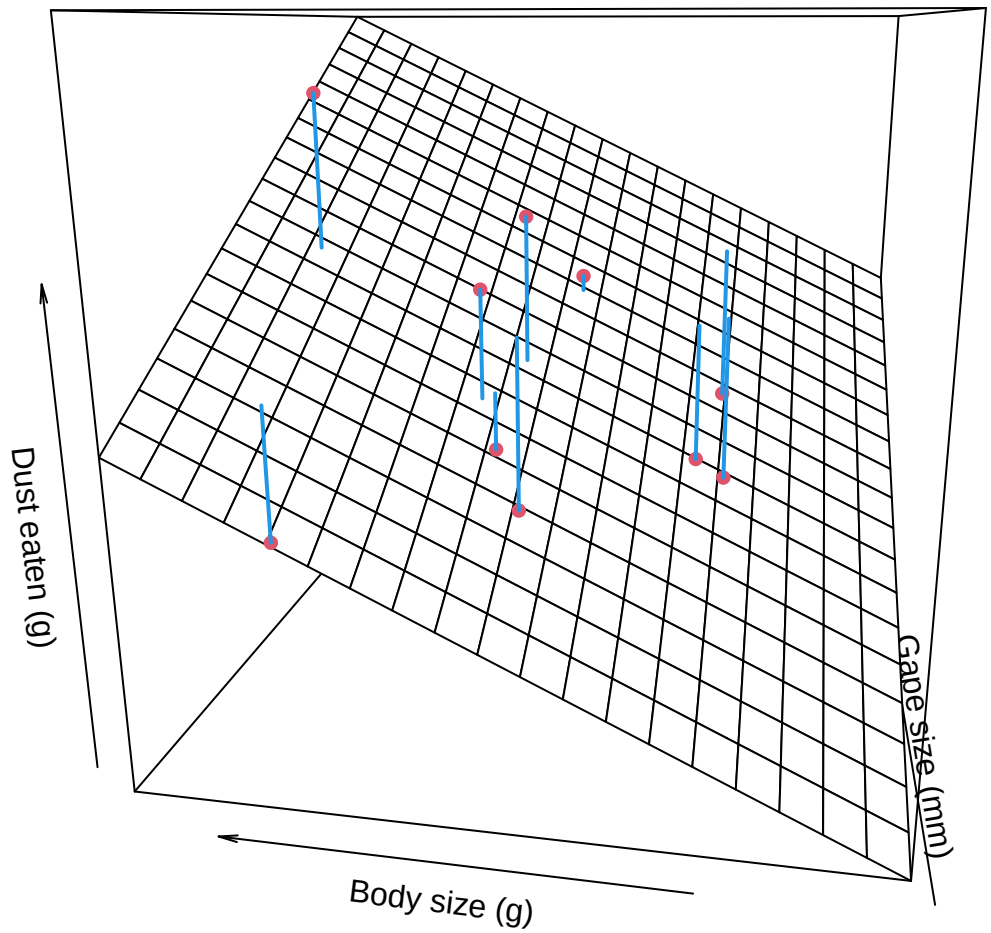
This is our model for simple regression

$$y_i = \alpha + \beta x_i + \varepsilon_i$$

The obvious extension to two covriates is to simply add on the effect of the second covariate

$$E(y_i) = \alpha + \beta_1 x_{1i} + \beta_2 x_{2i}$$

This equation is for a plane (which is just a line in 3D, of course). We can visualise the model (with a bit of difficulty). The plane is the black grid, the data are the red dots, which are either above or below the plane. The blue lines are the residuals: they project the points onto the plane, so we can see where their expected values are.



In many ways this model is almost the same as a simple regression. We have a fitted part, which is just a bit more complicated than before, and residuals, which are the same as before. And in other ways the models are the same, and we can use the same tools on them (e.g. model checking can be done the same way). Indeed, we use the same R function to fit the model (the maths is explained in more detail below):

```
FullMod <- lm(Dust ~ GapeSize + BodySize, data=Schey)
summary(FullMod)
```

The only change is in the formula. It was

```
Y ~ X
```

now it is

```
Y ~ X1 + X2
```

Exercise

For the data:

- first fit the model with each covariate individually (i.e. first explain dust eaten by gape size, then explain dust eaten by body size).
  - use `summary()` to look at the parameter estimates and R^2 (the Multiple R-squared: ignore the adjusted R^2). Write down the regression models (i.e. plug the correct values into $E(y_i) = \alpha + \beta_1 x_i$)
  - What do the models suggest are the effects on dust eating, and how well do the variables individually explain the variation in the response?
- fit a model with both covariates (i.e. explain dust eaten by both gape size and body size).

  - again, use `summary()` to look at the parameter estimates and R^2.Write down the regression model.
  - What does this model suggest are the effects on dust eating, and how well do the variables together explain the variation in the response?
  - How do these results compare to those from the single regression models?

Hint

The coefficients are in the `summary()` table, and the R^2 is at the bottom. There are several other statistics given, but ignore these (e.g. I've no idea why the Residuals are summarised).

For the interpretation, remember that a regression coefficient says that if we increase the covariate (i.e. the x) by one unit, the response (the y) changes by that amount.

Answers

- first fit the model with each covariate individually (i.e. first explain dust eaten by gape size, then explain dust eaten by body size).
  - use `summary()` to look at the parameter estimates and R^2. Write down the regression models (i.e. plug the correct values into $E(y_i) = \alpha + \beta_1 x_i$)
  - What do the models suggest are the effects on dust eating, and how well do the variables individually explain the variation in the response?

First, gape size:

```
##
## Call:
## lm(formula = Dust ~ GapeSize, data = Schey)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3824 -0.8788  0.0565  0.8573  2.7447
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  89.5042     3.1554  28.366  < 2e-16 ***
## GapeSize      3.6088     0.6311   5.718 1.17e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.098 on 98 degrees of freedom
## Multiple R-squared:  0.2502, Adjusted R-squared:  0.2425
## F-statistic: 32.69 on 1 and 98 DF,  p-value: 1.169e-07
```

So, reading from this, R^2 is 25%. And the model is

$$E(y_i) = 89.5 + 3.6x_i$$

i.e. the model explains about a quarter of the variation in dust eaten, and suggests an increase of 1mm in gape size means a Schey will eat (on average) 3.6g more dust.

Next, bodysize:

```
##
## Call:
## lm(formula = Dust ~ BodySize, data = Schey)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -3.4701 -0.8920  0.1478  0.8760  2.7279
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 104.5336     3.6719  28.468   <2e-16 ***
## BodySize      0.3000     0.3668   0.818    0.415
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.264 on 98 degrees of freedom
## Multiple R-squared:  0.00678,    Adjusted R-squared:  -0.003355
## F-statistic: 0.6689 on 1 and 98 DF,  p-value: 0.4154
```

So, reading from this, R^2 is 0.7%. And the model is

$$E(y_i) = 104.5 + 0.3x_i$$

i.e. the model explains almost nothing, and suggests an increase of 1g in body size means a Schey will eat (on average) 0.3g more dust.

Overall, the individual models explain some of the variation in the data, but not a huge amount, and suggest larger-mouthed Schey eat more dust.

- fit a model with both covariates (i.e. explain dust eaten by both gape size and body size).
  - again, use `summary()` to look at the parameter estimates and R^2.Write down the regression model.
  - What does this model suggest are the effects on dust eating, and how well do the variables together explain the variation in the response?
  - How do these results compare to those from the single regression models?

Let's fit the full model:

```
##
## Call:
## lm(formula = Dust ~ GapeSize + BodySize, data = Schey)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.15762 -0.32688  0.02161  0.34136  1.19986
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   9.3760     4.5203   2.074   0.0407 *
## GapeSize     10.6176     0.4761  22.300   <2e-16 ***
## BodySize      4.5092     0.2405  18.752   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5133 on 97 degrees of freedom
## Multiple R-squared:  0.8379, Adjusted R-squared:  0.8345
## F-statistic: 250.7 on 2 and 97 DF,  p-value: < 2.2e-16
```

R^2 for the model with both gape and body size is 83.8%. And the model is

$$E(y_i) = 9.4 + 10.6x_{1i} + 4.5x_{2i}$$

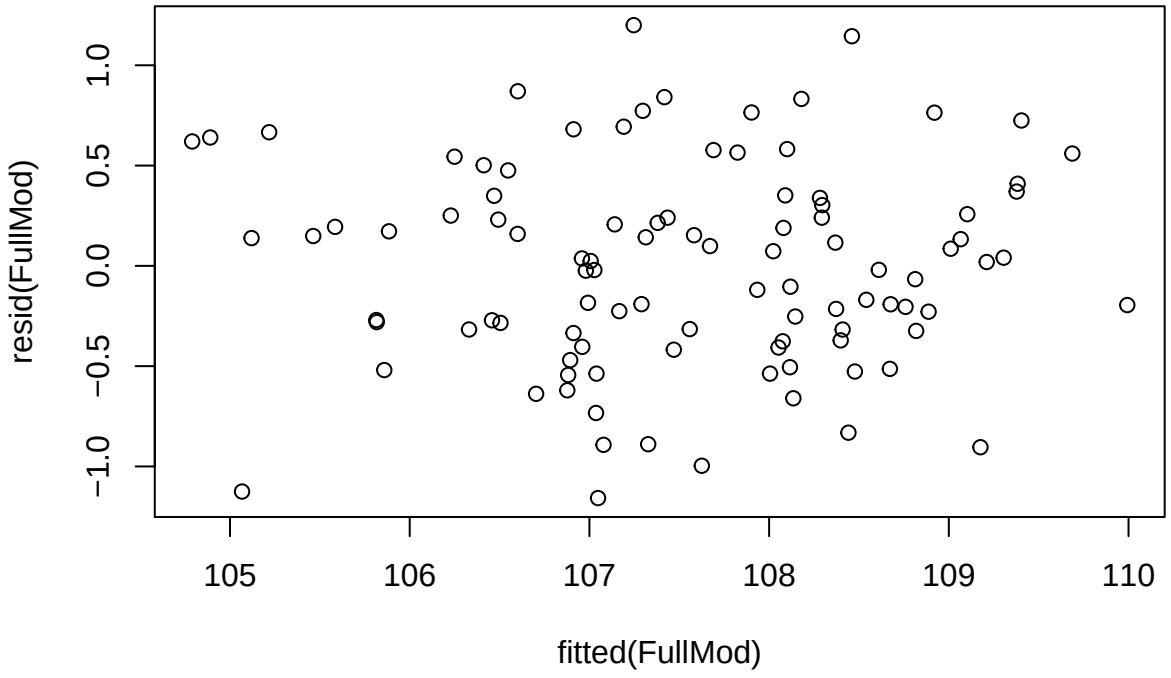
So, to compare. First, the R^2 is now much larger, the variables together explain over three quarters of the variation in the data, so we now have a good model.

The predicted effects are much larger too. Before, we predicted an increase of 1mm in gape size would increase the amount of dust eaten by 3.6g. But now we predict an increase of 10.6g. Similarly for body size, before we predicted an increase of almost nothing, 0.3g. But now we predict an increase of 4.5g when body size increases by 1g.

Hopefully this exercise has shown you that you can easily fit a multiple regression, and interpret the results. But a deeper point is adding covariates can change the whole model, not just the new covariate. In this case, the negative correlation between gape size and body size masked the large effects of each variable.

We can also use the same tools as we used in the univariate model, e.g. `coef()` to get thh parameter estiamtes (of course there are now more of them), and `resid()` and `fitted()` to look at the model fit:

```
## (Intercept)    GapeSize    BodySize
##    9.376014   10.617635    4.509229
```



Regression More Generally

The model above only has 2 covariates, but we can easily add more. The model will look like this

$$y_i = \alpha + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \cdots + \beta_p x_{ip} + \varepsilon_i$$

$$y_i = \alpha + \sum_{j=1}^{p} \beta_j x_{ij} + \varepsilon_i$$

- we have p covariates, labelled from $j = 1$ to p
- we have p covariate effects
- the jth covariate values for the ith individual is x_{ij}

Obviously we can add as many covariates as we want, although the model will not fit if there are more covariates than data points, and in practice we would like to have far fewer covariates, becuse each covariate makes the model a little less certain.

Writing the model like this can get messy, espacially if we want to manipulate it. But we can write it as a matrix. This is, in some ways, a detail, but the practical upshot is that we can work with the matrix formulation to find out how to fit the model, and then for any complicated model we just have to be able to write it in this matrix form, and everything else just follows.

The first step in writing this as amatrix is to turn the intercept into a covariate by using a covariate with a value of 1 for every data point. Then we write all of the covariates in a matrix, X.

$$X = \begin{pmatrix} 1 & 2.3 & 3.0 \\ 1 & 4.9 & -5.3 \\ 1 & 1.6 & -0.7 \\ \vdots & \vdots & \vdots \\ 1 & 8.4 & 1.2 \end{pmatrix}$$

The first column is the intercept, the second is the first covariate, and the third is the second covariate. This is called the *Design Matrix*. Using matrix algebra, the regression model becomes

$$\mathbf{Y} = X\beta + \varepsilon$$

where $\mathbf{Y}$, β and ε are now all vectors of length n, where there are n data points. X is an $n \times (p+1)$ matrix. We will not look at the mathematics in any detail, the point here is that the model for the effect of covariates can be written in the design matrix. It turns out that this is very flexible, if we have more covariates, or interactions betwen covariates, we can still write them in a design matrix. The model, in all its ugly glory, is

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & 2.3 & 3.0 \\ 1 & 4.9 & -5.3 \\ 1 & 1.6 & -0.7 \\ \vdots & \vdots & \vdots \\ 1 & 8.4 & 1.2 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \varepsilon_n \end{pmatrix}$$

After a bit of matrix algebra, one can find the maximum likelihood solution

$$\mathbf{b} = (X^T X)^{-1} X^T \mathbf{Y}$$

where $\mathbf{b}$ is the MLE for β. We won't show you the proof, and you won't need to remember this. In practice the computer will do all the calculations for you.

Interpreting the Model

In the exercise you were asked to interpret the coefficients by comparing the coefficients for the same covariate in different models. But this is only one problem that we might need to think about when interpreting what we've done. When we have a model with, say 10, covariates, how do we know which is the modet important? Unfortunately we can't just compare the coefficients because they are measured on different scales. How does a change in weight of 1g compare to a change in size of 1mm? And the measurements may even be on different scales: a change of 1mm is different to 1m, but both are one unit.

This is one motivation for what we are about to do, after which we will see another motivation. And then next week, when we look at categorical variables, some of the ideas will appear again. We are going to look at re-writing the model in different ways by adding and multiplying. Although the model is essentially the same, and it is easy to go between one model and another, some models are easier to interpret.

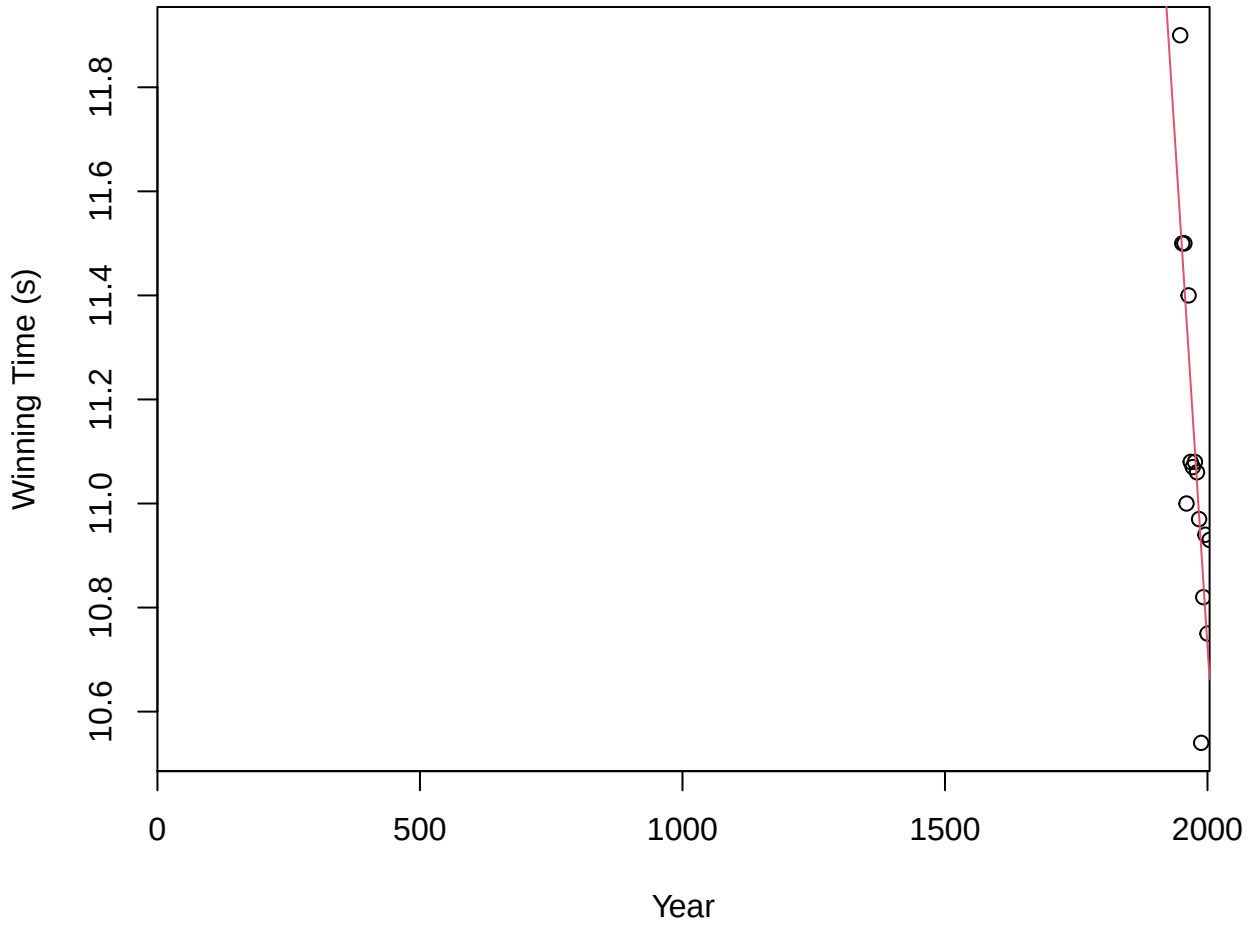
Mean Centering

So far we have not been thinking about the intercept. It is a necessary part of the model, but is not so easy to interpret.

The intercept is where the fitted line crosses the point where the covariate(s) equal 0 (because the prediction for there is $E(y_i) = \alpha + \beta_1 0 + \beta_2 0 = \alpha$). The interpretation is difficult, because we cannot have any Schey with a size and gape of 0, so as a prediction this is silly. That's fine as long we don't want to interpret the intercept.

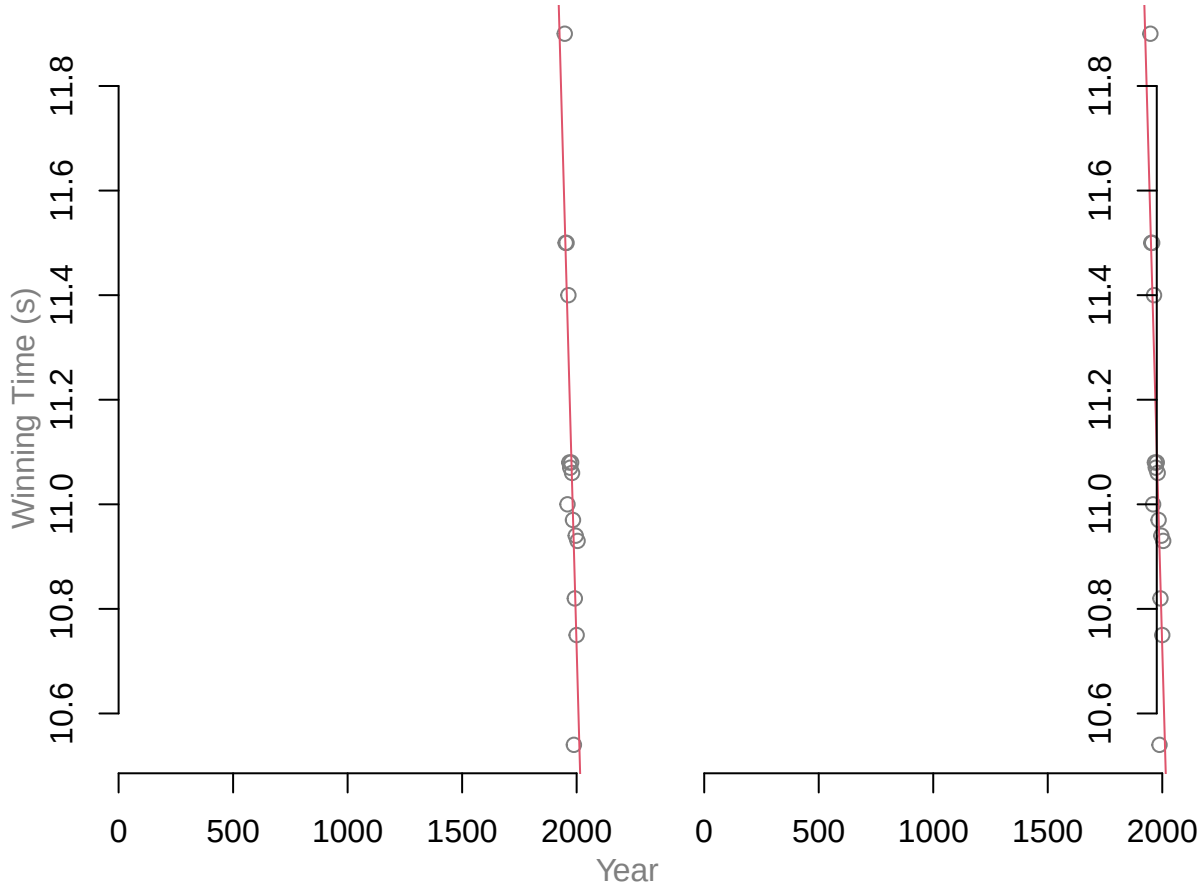
We can look a the Women's 100m data we have been using before. Here we plot the data with the intercept ($x = 0$) included.

```
Times <- read.csv('https://www.math.ntnu.no/emner/ST2304/2019v/Week5/Times.csv', header=T)
OlympicModelW <- lm(WomenTimes ~ Year, data  = Times)

par(mar=c(4.1,4.1,1,1))
plot(Times$Year, Times$WomenTimes, xlim=range(c(0, Times$Year)), xaxs="i",
     xlab="Year", ylab="Winning Time (s)")
abline(OlympicModelW, col=2)
```



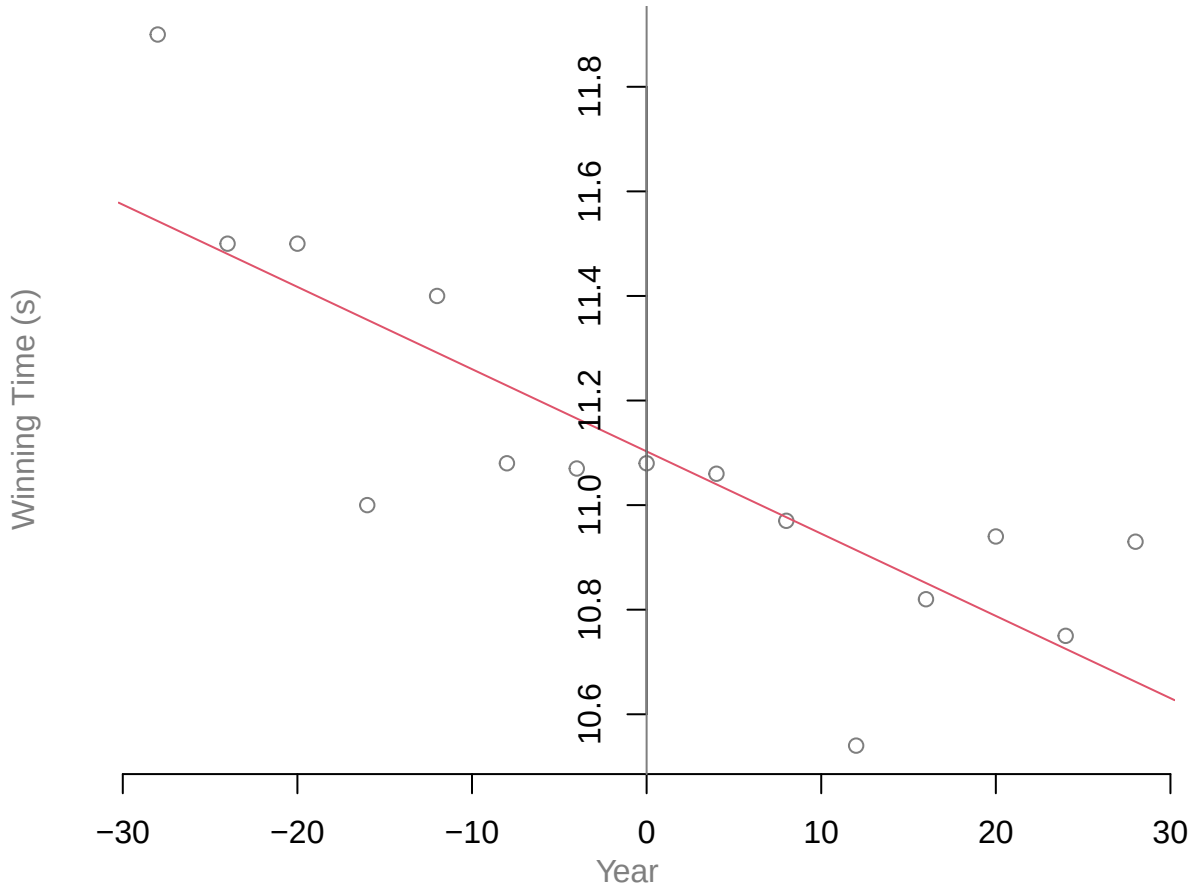
This looks a bit silly, and the suggestion that at 0 AD the winning time would be an unlikely 42s. This is not usuallly a problem as long as we don't care about the intercept. But sometimes we can find it helpful to have an intercept that is interpretable, if only so we can do a sanity check. We can easíly move the intercept like this:

```
par(mfrow=c(1,2), mar=c(2,2,1,1), oma=c(2,2,0,0), bty="n", col="grey50")
plot(Times$Year, Times$WomenTimes, xlim=range(c(0, Times$Year+50)),
     xaxs="i", yaxt="n", xlab="", ylab="")
abline(OlympicModelW, col=2)
axis(2, pos=0)

plot(Times$Year, Times$WomenTimes, xlim=range(c(0, Times$Year+50)), xaxs="i",
     yaxt="n", xlab="", ylab="")
abline(OlympicModelW, col=2)
axis(2, pos=mean(Times$Year))
mtext("Year", 1, outer=TRUE)
mtext("Winning Time (s)", 2, outer=TRUE)
```



In practice this just means subtracting the mean from the x (i.e. year)

```
Times$YearCentred <- Times$Year - mean(Times$Year)
OlympicModelWCentred <- lm(WomenTimes ~ YearCentred, data  = Times)

par(mfrow=c(1,1), mar=c(2,2,1,1), oma=c(2,2,0,0), bty="n", col="grey50")
plot(Times$YearCentred, Times$WomenTimes,
     yaxt="n", xlab="", ylab="")
abline(OlympicModelWCentred, col=2)
axis(2, pos=0); abline(v=0)
mtext("Year", 1, outer=TRUE)
mtext("Winning Time (s)", 2, outer=TRUE)
```



The intercept is now 11.1s, which is the predicted time for the year 1976. As a sanity check, this is reasonable.

Exercise

- For the Schey data, fit the models with the un-centred and centred Body Size and Gape Size. Look at the parameters (with `coef()`), and discuss any differences. Look at both the intercept and the slopes.
- Can you interpret the parameters?

Hint

This is the code to calculate the centred body size and fit the model.

```
Schey$BodySize.c <- Schey$BodySize - mean(Schey$BodySize)

FullMod.c <- lm(Dust ~ GapeSize.c + BodySize.c,
                data=Schey)
coef(FullMod.c)
```

Answers

This is the code to calculate the centred body size and fit the model.

```
Schey$BodySize.c <- Schey$BodySize - mean(Schey$BodySize)
Schey$GapeSize.c <- Schey$GapeSize - mean(Schey$GapeSize)

FullMod.c <- lm(Dust ~ GapeSize.c + BodySize.c,
                data=Schey)
FullMod <- lm(Dust ~ GapeSize + BodySize,
                data=Schey)
```

```
coef(FullMod)
```

```
## (Intercept)    GapeSize    BodySize
##    9.376014   10.617635    4.509229
```

```
coef(FullMod.c)
```

```
## (Intercept)  GapeSize.c  BodySize.c
##  107.535025   10.617635    4.509229
```

The easy thing to notice is that the slopes - the Gape Size and Body Size effects - are the same. This is what we expect to see. But the intercept was 9.4g and after centering it is 107.5g. This is within the range of the observations of dust eaten (103.9g - 110.2g), and actually equals the mean, 107.5.

Scaling and Standardisation

Body size was measure in grams, but it could also be measured in kg. If we fit the model with this, we see that the effect of body size is massive.

```
Schey$BodySize.kg <- Schey$BodySize/1000
mod.kg <- lm(Dust ~ GapeSize + BodySize.kg, data=Schey)

round(coef(mod.kg), 2)
```

```
## (Intercept)    GapeSize BodySize.kg
##        9.38       10.62     4509.23
```

Exercise

We want to you to think about this, and discuss the parameters, and what they mean.

- can you interpret the slopes in terms of predictions?
- Why is the effect so massive?

Hint

How do you interpret the regression coefficients? They say something about the change in Dust when body size changes, but can you say what?

Answers

The slope tells you how much the dust eaten changes with a change in one unit of the covriate, i.e. body mass. So here it says that if the body mass increased by 1kg, we would expect 4.51kg more dust to be eaten. But as the largest Schey we caught only weighed 10.7g (or 0.0107kg) this amount of change seems unlkely.

Standardisation

Because we can re-scale the parameters, and still do the regression, we can (if we want) re-scale to anything we think is sensible. For example, we could swap temperatures between Kelvin, Celcius and Fahrenheit depending on our whims. One way of re-scaling them that is often used is to standardise them to have a variance (and standard deviation) of 1. Here are two ways of doing this in R, the first does it "by hand", the second uses an R function. But both do the same thing.

```
Schey$BodySize.s <- (Schey$BodySize - mean(Schey$BodySize))/
  sd(Schey$BodySize)
Schey$GapeSize.s <- scale(Schey$GapeSize)
```

Now a difference of 1 mean a difference in 1 standard deviation in the data (e.g. 1 standard deviation in body size, rather tha 1g). But internally the model is the same [1], although the parameters have different values and thus have to be interpreted slightly differently.

```
FullMod.s <- lm(Dust ~ GapeSize.s + BodySize.s,
                data=Schey)
round(coef(FullMod.s), 3)
```

Why, you may be wondering, is this useful? The reason is that it can make the estimates more comparable. By standardising like this, we have one way to compare the effects of different covariates. Our argument is that the effect of gape size is the effect of changing gape size by one standard deviation, and the effect of body size is the effect of changing body size by one standard deviation.

So in some sense these are comparable. If we randomly sampled our Schey from the population, the standardised coefficients then say something about the variation in the population. This does require that we are sampling randomly from the population, so (for example) if we have an experiment where we set the different levels we can't do this as the standard deviations are not the same as the population-level standard deviations. e.g. if we wanted to look at the effects of temperature and humidity on dust eating, we might set the temperature to 10°C, 15°C, 20°C, 25°C and humidity to 50%, 60%, 70%, 80%. But these might not have any relationship to the distribution of actual temperatures and humidities.

Exercise

Fit the model with the standardised coefficients

- How would you interpret the standardised coefficients?
- When might you prefer to use the standardised or un-standardised models?

Hint

Think about how much a change in one unit of the covariate means.

Answers

```
Schey$BodySize.s <- (Schey$BodySize - mean(Schey$BodySize))/
  sd(Schey$BodySize)
Schey$GapeSize.s <- scale(Schey$GapeSize)


mod <- lm(Dust ~ GapeSize + BodySize, data=Schey)
mod.s <- lm(Dust ~ GapeSize.s + BodySize.s, data=Schey)

round(coef(mod), 2)
```

```
## (Intercept)    GapeSize    BodySize
##        9.38       10.62        4.51
```

```
round(coef(mod.s), 2)
```

```
## (Intercept)  GapeSize.s  BodySize.s
##      107.54        1.86        1.56
```

[1] A quick bit of maths. The standardised model (for one variable) is

$$y_i = \alpha + \beta \frac{(x_i - \bar{x})}{s_x} + \varepsilon_i$$

where $\bar{x}$ is the mean of x and s_x is the standard deviation of x. We can expand the brackets and re-arrange to get

$$y_i = \alpha + \beta x_i / s_x - \beta \bar{x} / s_x + \varepsilon_i$$

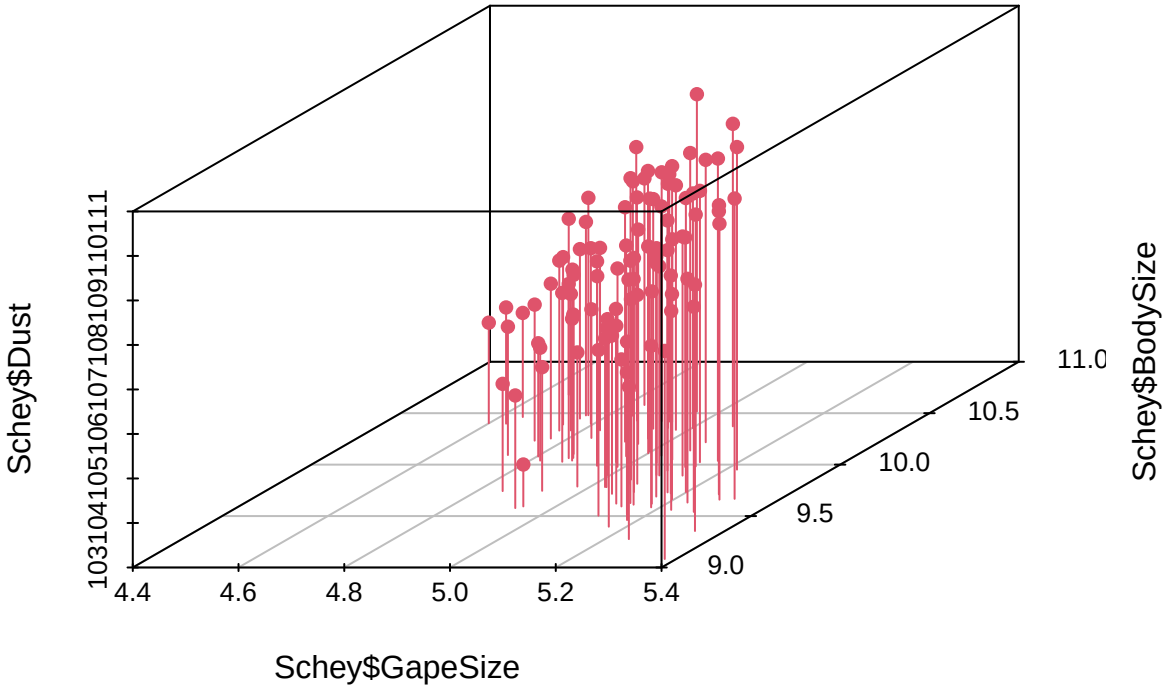
But $\bar{x}_{.j}$ is a constant - it does not vary for different y's, so we have the same model, but with $\alpha^* = \alpha - \frac{\beta_j}{s_j}\bar{x}_{.j}$ and $\beta_j^* = \frac{\beta_j}{s_j}$

The first coefficients are for the unstandardised data, and the coefficient for Gape Size is larger. But when we look at the standardised coefficients, they are about the same - 1.86 and 1.56. This suggests that in the population, the variation in gape size and body size have about the same effect on the amount of dust eaten.

For the comprison in the previous paragraph, the standardised coefficients make more sense. But if we were working with Schey and measuring them in the field to see how much they could eat, we would probably prefer to use the unstandardised model, because it is easier to do the calculations. You can probably come up with other reasons to prefer one or the other (and "I prefer the unstandardised because I understand what it means" is a perfectly valid reason!).

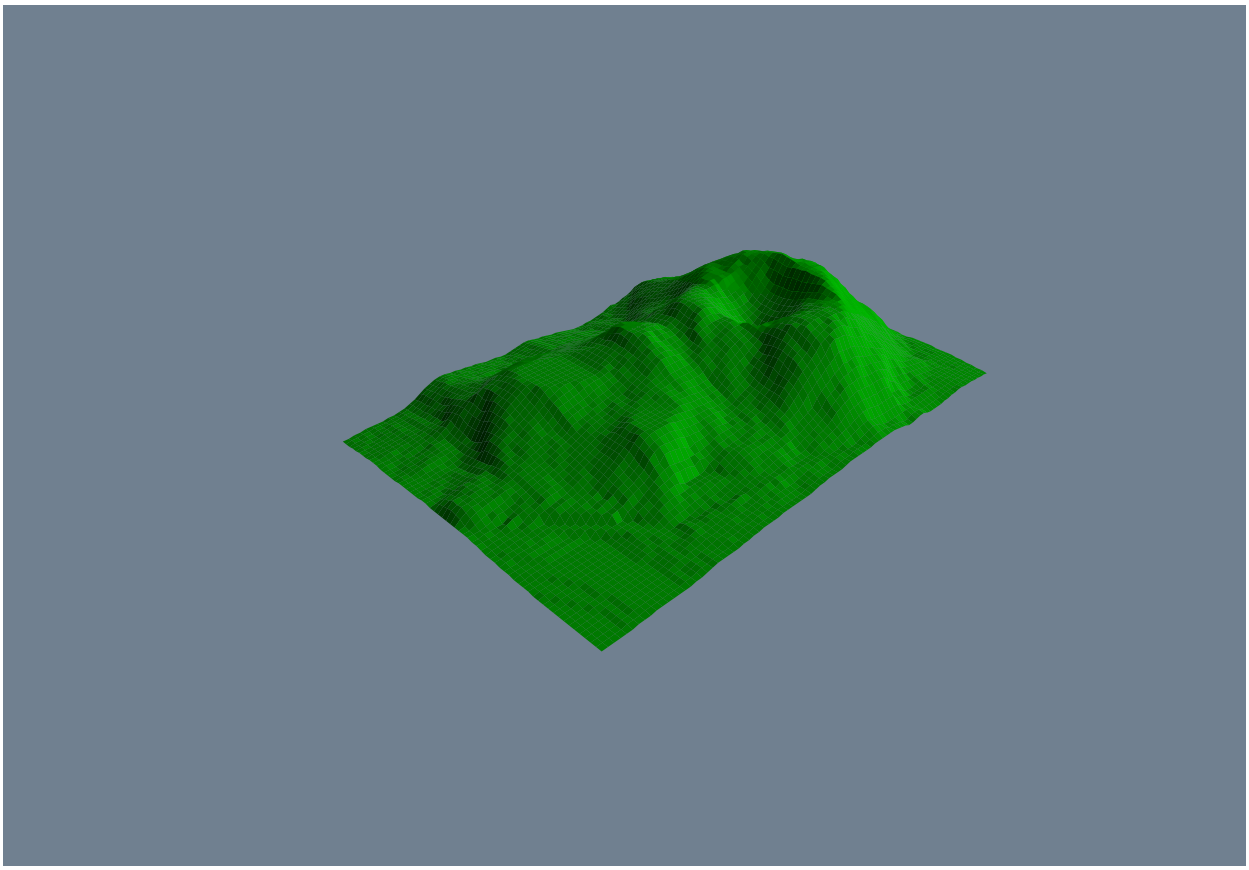
Extending the Model: Fitting Surface

The plot below shows the data in 3D, and the line fitted for the model with body size.



We can see that the data form a plane. If we have more than 2 covariates, then we get a hyper-plane, which is more difficult to visualise.

For other problems, we might have a more complicted surface than a plane. For example this surface:



A plane will obviously not work to describe this surface, so we need something more complicated. There are a lot of different approaches to this now: essentially all of machine leaning in about fitting surfaces to data. Mathematically we want a model like this:

$$y_i = f(X_i) + \varepsilon_i$$

where X_i can be several covariates, and $f()$ is some function. One common way to do this is to keep the linear regression model idea, and transform the X_i's:

$$y_i = \sum_{j=1}^{P} \beta_j g(X_{ij}) + \varepsilon_i$$

Before P was the number of covariates, now it is the number of "features", i.e. the number of $g()$'s. We will see in a moment what these could look like. So now we have a multiple regression of y_i against $g(X_{ij})$. Which is fine, but how do we chose $g()$? Here we'll explain one way, which first needs a short excursion into some old mathematics.

Polynomials

In 1715 an English mathematician called Brook Taylor showed that any well enough behaved mathematical function can be approximated by a polynomial:

$$f(x, a) = \sum_{j=0}^{\infty} \frac{f^{(j)}(a)}{j!}(x - a)^j$$

So the polymonial has terms Ax (linear), Bx^2 (squared, or quadratic), Cx^3 (cubed) etc. This is called a *Taylor series expansion* around a: it provides an approximation for the curve around the point a, and as we add higher values of j to the sum, the approximation gets better (and is better further away from a). In case you are wondering, a is there because we want to approximate the function near that point: the further away from a, the worse the approximation. You may already be able to guess what we are going to use for a.

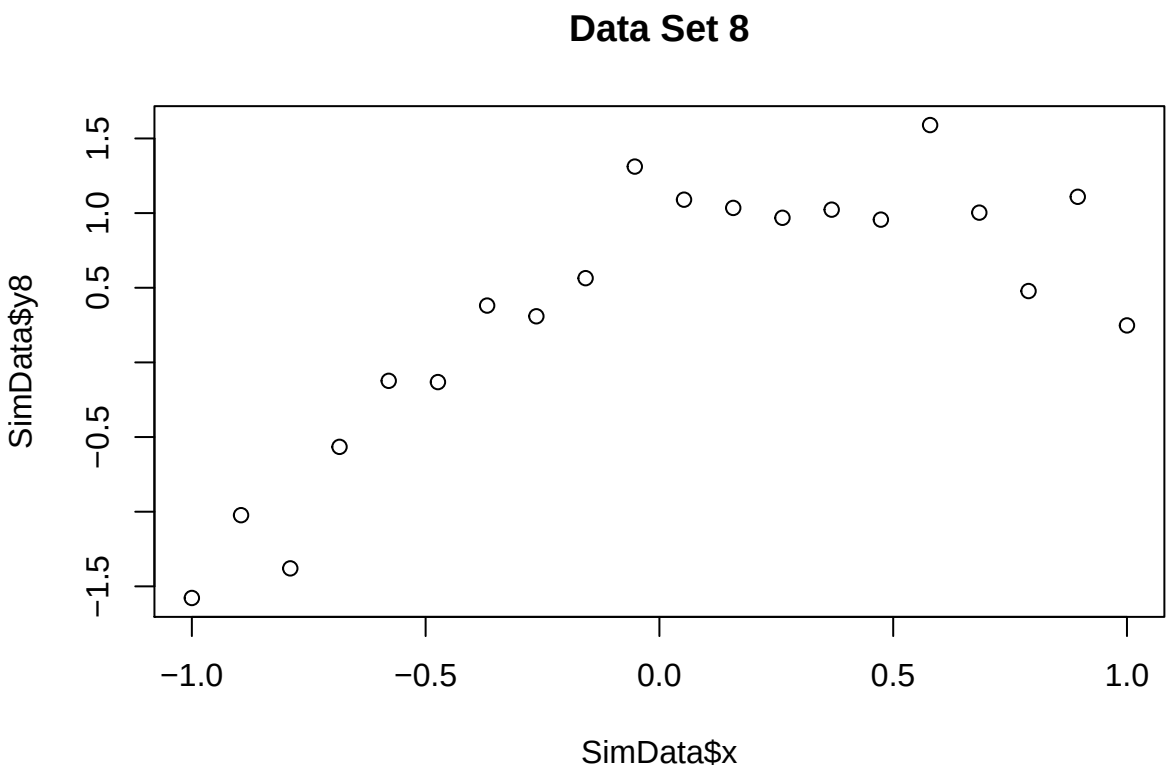
For our purposes, we can re-write the function to look like a multiple regression, and not use all of the powers (i.e. we don't go up to x^∞):

$$E(x|\mu) = \sum_{j=0}^{P} \beta_j (x-\mu)^j = \beta_0 + \beta_1(x-\mu) + \beta_2(x-\mu)^2 + ... + \beta_P(x-\mu)^P$$

So this is a P^{th} order Taylor Series expansion around μ, i.e. we centre the covariate and we only use the powers up to x^P. We can then regress our y against powers of the covariate. We try to keep P fairly small (if it goes above about 3 or 4 there is proabably a better model). Going back to the curve fitting idea, here the $g()$'s are simply powers of x.

We can look back to Data Set 8 last week to see how to do this in practice.

```
SimData <- read.csv("https://www.math.ntnu.no/emner/ST2304/2019v/Week6/SimRegression.csv")
plot(SimData$x, SimData$y8, main="Data Set 8")
```



Fitting the curve is easy, we just regress Y against X, X^2, X^3 etc. We don't have to centre (see footnote 1 below for why), although it might make the interpetation easier.

In R we can simply treat the extra terms as additional variables:

```
linmod <- lm(y8 ~ x, data=SimData)
quadmod <- lm(y8 ~ x + I(x^2), data=SimData)
```

We need to write `I(x^2)` rather than just `x^2` for slightly obscure reasons, to do with writing more complex models[2].

If we want to plot a polynomial, we can't use `abline()`, unfortunately. Instead we have to predict new data, and plot that.

```
PredData <- data.frame(x=seq(min(SimData$x),
                             max(SimData$x), length=50))
PredData$y.quad <- predict(quadmod, newdata = PredData)

linmod <- lm(y8 ~ x, data=SimData)
cubmod <- lm(y8 ~ x + I(x^2) + I(x^3), data=SimData)

PredData$y.lin <- predict(quadmod, newdata = PredData, interval = "predict")
PredData$y.quad <- predict(cubmod, newdata = PredData, interval = "predict")

plot(SimData$x, SimData$y8, main="Data Set 8")
lines(PredData$x, PredData$y.quad, col=2)
```

Exercise

Your task is to fit the linear and quadratic models to `y8`.

- Does the quadratic model fit better?
- Are the parameters different?
- What happens if you add an x^3 term?
- Plot the curves. How different are they? Just by looking at the plots, which do you think is best, and why?

Hint

You can use and adapt the code above. What other statistics could you use to look at model fit?

Answers

- Does the quadratic model fit better?
- Are the parameters different?
- What happens if you add an x^3 term?
- Plot the curves. How different are they? Just by looking at the plots, which do you think is best, and why?

First, fit the linear, quadratic and cubic model:

and then look at the parameters:

```
## (Intercept)           x
##   0.3632011   1.1201099
```

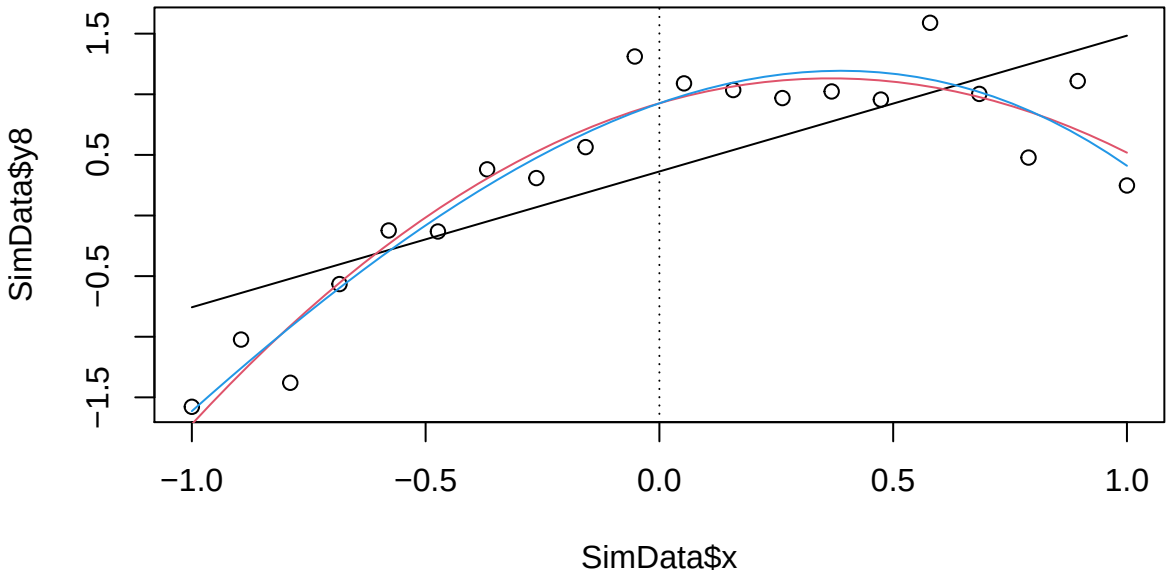
```
## (Intercept)           x      I(x^2)
##   0.9260229   1.1201099  -1.5276592
```

```
## (Intercept)           x      I(x^2)      I(x^3)
##   0.9260229   1.3309775  -1.5276592  -0.3190410
```

We see that the parameters change as the model changes, e.g. the intercapt increases when we add the quadratic term. Also note that the quadratic term is negative, i.e. the slope becomes smaller as x increases.

We can get a better feel for what is going on if we plot the curves:

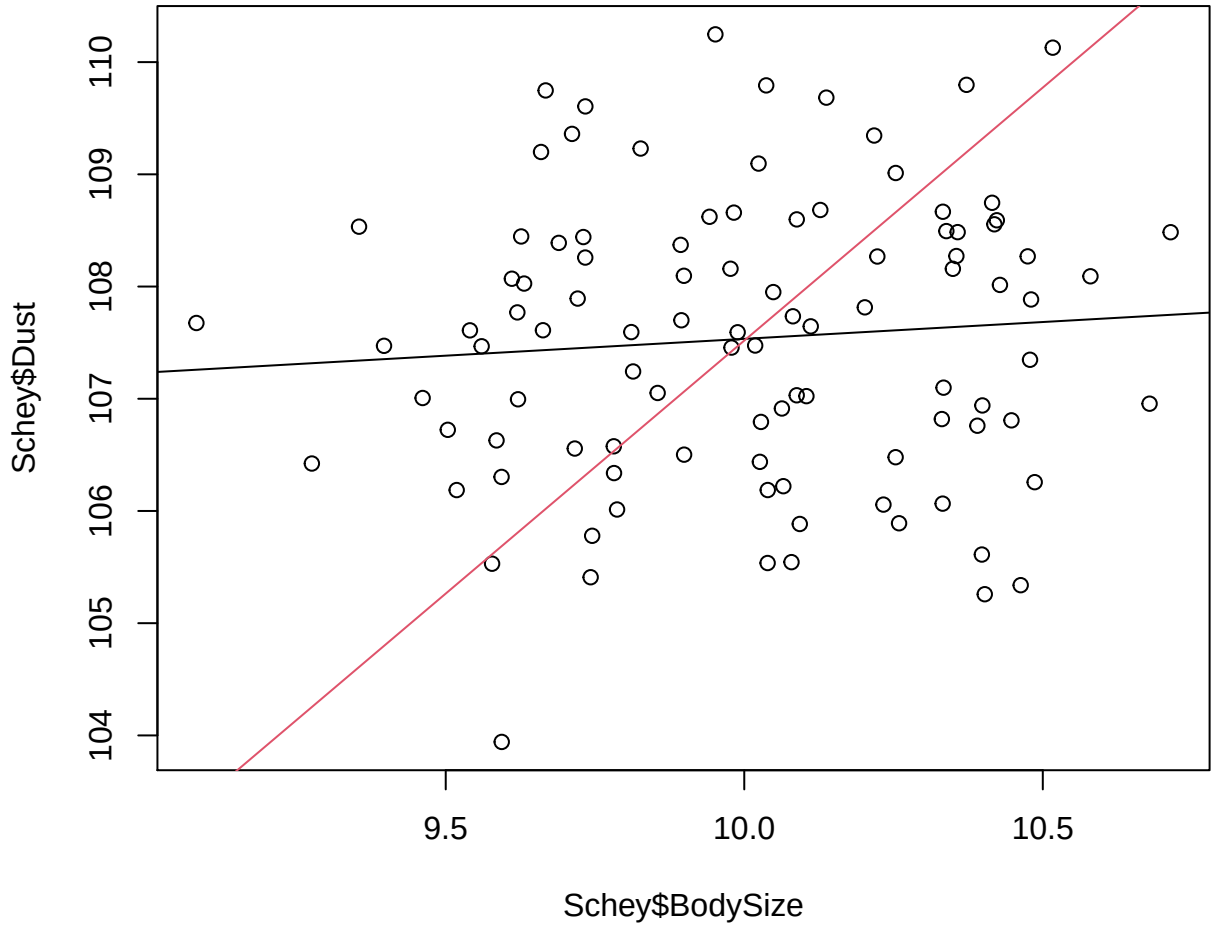
[2] essentially, we want `(A + b)^2 = A + A:B + B`. We will explain more about this later



So we can see that the linear curve is a poor fit, but the quadratic and cubic models both fit well, and are not very different. It is also worth looking at R^2. This is 59.1% for the linear model, 91.2% for the quadratic and 91.6% for the cubic. So the linaer model fits quite a bit worse than the other two, and these have a very similar fit (the cubic will always fit better than the quadratic, because it is more flexible). But a difference of 0.4% is almost nothing.

If we want to select a single line to draw, the mean of gape size is a good choice

```
Better.a <- coef(FullMod)["(Intercept)"] +
  coef(FullMod)["GapeSize"]*mean(Schey$GapeSize)
par(mar=c(4.1,4.1,1,1))
plot(Schey$BodySize, Schey$Dust)
abline(a=coef(BodyMod)["(Intercept)"],
       b = coef(BodyMod)["BodySize"])
abline(a=Better.a, b = coef(FullMod)["BodySize"], col=2)
```


Schey$Dust
110
109
108
107
106
105
104
9.5
10.0
10.5
Schey$BodySize

—>