Applied Statistics (MATH10096)

Simon Wood & Vanda Inácio

Contents

1 Preliminaries

This course builds on Statistics (Year 2) (MATH08051) and on Statistical Methodology (MATH10095). In particular it assumes that you are familiar with the key ideas behind parametric statistical inference.

- *Statistics* is about using data to extract meaningful information about the system generating the data, when the data generating process is such that the data will vary randomly from one replication of the data generating process to the next, even if the underlying system stays the same.
- A *statistical model* is a simplified mathematical representation of the data generating process (a sort of mathematical cartoon of the system that we want to learn about). It will usually depend on some known things, such as other variables we have measured alongside the data, and some unknown parameters: $\boldsymbol{\beta}$, say. At this point, we quote Sir David Cox, in a comment in the *Journal of the Royal Statistical Society, Ser. A*, **158**, 455–456 (1995): "*The very word model implies simplification and idealization. The idea that complex physical, biological or sociological systems can be exactly described by a few formulae is patently absurd. The construction of idealized representations that capture important stable aspects of such systems is, however, a vital part of general scientific analysis.*"
- A key point about a statistical model is that *if* we knew the values of $\boldsymbol{\beta}$, then the model should be able to simulate data that 'looked like' the real data. In principle, given values for the parameters, a statistical model also allows us to assign relative probabilities to observing one data set, as opposed to another.
- Once we have a model, the major goal of statistics is then to make *inferences* about $\boldsymbol{\beta}$ from the data. There are four questions:
  1. What value of $\boldsymbol{\beta}$ is most consistent with the data?
  2. What range of values of $\boldsymbol{\beta}$ is consistent with the data
  3. Is some pre-specified value of, or restriction on, $\boldsymbol{\beta}$ consistent with the data?
  4. Are there any values of $\boldsymbol{\beta}$ at all for which the model is consistent with the data?
- The answers to these questions are provided by
  1. Parameter estimation, especially maximum likelihood estimation, which seeks to find the $\hat{\boldsymbol{\beta}}$ making the observed data as probable as possible, according to the model.
  2. Confidence interval estimation, which uses the data to compute intervals with a specified probability of including the true parameter values, over replication of the data generating process .
  3. Hypothesis testing, which seeks to assess the plausibility of some hypothesis, by computing a measure of how improbable the data are under that hypothesis: the *p-value* (see later). A low p-values suggests that the data would be improbable if the hypothesis were true, so it's probably false.
  4. Model checking: could the model have produced the data at all? If it could not then none of the theory in 1-3 is of any use at all. Useful checking is often done graphically.

In terms of technicalities, the course particularly assumes that you are familiar with random variables, p.d.f.s, c.d.f.s and inverse c.d.f.s (quantile functions). It also assumes that if you are told $\hat{\beta} \sim N(\beta, \sigma^2_{\hat{\beta}})$ or $(\hat{\beta} - \beta)/\hat{\sigma}_{\hat{\beta}} \sim t_d$ (plus any values needed), then you could test hypotheses about β and find confidence intervals for β.

Topics to be covered in this course include:

- Thorough, but quick, **review of linear models** (LMs).
- **Generalised linear models** (GLMs): a *generalisation* of linear regression models that allows for the response variable to have a distribution other than the normal by modelling a transformed expectation of the response variable. This class of models is able to handle, for example:

  - *Binary* or *categorical* response variables: in cases where the outcome is binary (e.g., success/ failure, yes/no) a logistic regression model, which is a type of GLM, can be used. When the response variable has a number (> 2) of categories that do not have any order to them (e.g., modelling vote choice in elections), multinomial logistic regression may be used.
  - *Count* response variables: when the outcome variable is a count (e.g., number of car deaths in a given year), Poisson regression, another type of GLM, is often appropriate.
  - *non-negative* continuous response variables: when the outcome variable is continuous but non-negative (e.g., size of claims made by its policyholders), a gamma regression model, also a particular case of a GLM, can be an option.

- **Mixed effects models**: LMs and GLMs assume random variability is independent between observations. However, in many real-world applications, observations are not independent of each other. For instance, in education research, students within the same school or classroom may perform similarly due to shared environment or teaching methods. Mixed models by including the so-called *random effects*, which introduce a source of variation into the model beyond fixed effects (the usual regression coefficients), are capable of handling non-independent observations.

- **Generalised additive models** (GAMS) (if time permits): allow for *nonlinear smooth* relationships between the response variable and one or more covariates.

Useful books

There are dozens of books on regression models, ranging from those with a more theoretical focus to others that are more applied. No specific textbook is required for this module, as the lecture notes are self-contained, assuming you have the necessary prior knowledge. However, below you can find a list of some interesting books (by alphabetical order of the first author). Please note that these books are not mandatory, and they might not cover all the topics we will learn, or they may include additional material.

- Dobson, A. J., Barnett, A. G. (2018). *An Introduction to Generalized Linear Models*, 4th edition, CRC Press.
- Fahrmeir, L., Kneib, T., Lang, S., and Marx, B. D. (2022). *Regression: Models, Methods and Applications*, 2nd edition, Springer.
- Faraway, J. (2014). *Linear Models with R*, 2nd edition, CRC Press.
- Faraway, J. (2016). *Extending the Linear Model with R: Generalized Linear, Mixed Effects, and Nonparametric Regression*, 2nd edition, CRC Press.
- Gelman, A., Hill, J., and Vehtari, A. (2020). *Regression and Other Stories*, Cambridge University Press.
- McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*, 2nd edition, Chapman & Hall/CRC.
- Roback, P., and Legler, J. (2020). *Beyond Multiple Linear Regression: Applied Generalized Linear Models And Multilevel Models in R*, CRC Press.
- Wood, S. N. (2017). *Generalized Additive Models: An Introduction with R*, 2nd edition, CRC Press.

A substantial portion of the material in these lecture notes is based on the final reference cited.

2 Introduction

How old is the universe? The standard big-bang model of the origin of the universe says that it expands uniformly, and locally, according to Hubble's law,

$$y = \beta x,$$

where y is the recessional velocity of a galaxy (i.e., how fast it is moving away from the observer) at a distance x (from Earth), and β is "Hubble's constant" (in standard astrophysical notation $y \equiv v$, $x \equiv d$ and $\beta \equiv H_0$). β^{-1} gives the approximate age of the universe, but β is unknown and must somehow be estimated from observations of y and x, made for a variety of galaxies at different distances from us. The following link provides a nice explanation of the Hubble's law:

https://news.uchicago.edu/explainer/hubble-constant-explained

Of course, there will always be Wikipedia as well

https://en.wikipedia.org/wiki/Hubble

Figure 1 plots velocity against distance for 24 galaxies, according to measurements made using the Hubble Space Telescope. Velocities are assessed by measuring the Doppler effect red shift in the spectrum of light observed from the galaxies concerned, although some correction for 'local' velocity components is required. Distance measurement is much less direct, and is based on the 1912 discovery, by Henrietta Leavit, of a relationship between the period of a certain class of variable stars, known as the Cepheids, and their luminosity. The intensity of Cepheids varies regularly with a period of between 1.5 and something over 50 days, and the mean intensity increases predictably with period. This means that, if you can find a Cepheid, you can tell how far away it is, by comparing its apparent brightness to its period predicted intensity.

It is clear, from the figure, that the observed data do not follow Hubble's law exactly, but given the measurement process, it would be surprising if they did. Given the apparent variability, what can be inferred from these data? In particular:

(i) What value of β is most consistent with the data?

(ii) What range of β values is consistent with the data?

(iii) Are some particular, theoretically derived, values of β consistent with the data?

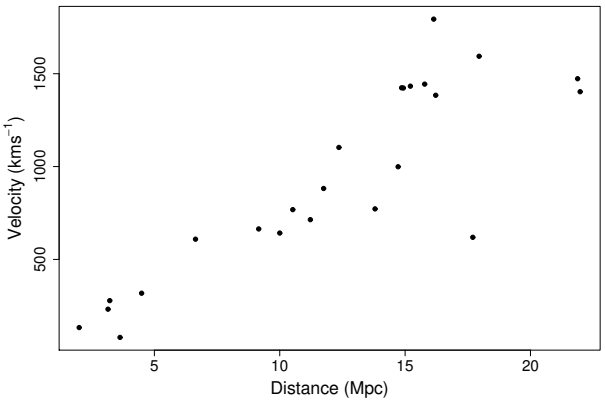


Figure 1: A Hubble diagram showing the relationship between distance, x (measured in megaparsecs), and velocity, y (measured in kilometers per second), for 24 galaxies containing Cepheid stars. The data are from the Hubble Space Telescope key project to measure the Hubble constant.

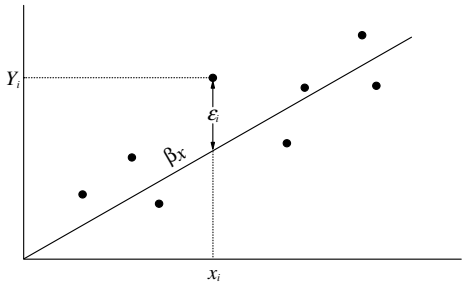


Figure 2: Schematic illustration of a simple linear model with one explanatory variable.

Statistics is about trying to answer these three sorts of questions.

One way to proceed is to formulate a linear statistical model of the way that the data were generated, and to use this as the basis for inference. Specifically, suppose that, rather than being governed directly by Hubble's law, the observed velocity is given by Hubble's constant multiplied by the observed distance plus a 'random variability' term. That is

$$y_i = \beta x_i + \epsilon_i, \quad i = 1 \dots 24, \tag{1}$$

where the ϵ_i terms are independent random variables such that $\mathbb{E}(\epsilon_i) = 0$ and $\mathbb{E}(\epsilon_i^2) = \sigma^2$, thus implying that $\text{var}(\epsilon_i) = \sigma^2$. The random component of the model is intended to capture the fact that if we gathered a replicate set of data, for a new set of galaxies, Hubble's law would not change, but the apparent random variation from it would be different, as a result of different measurement errors. Notice that it is not implied that these errors are completely unpredictable: their mean and variance are assumed to be fixed, it is only their particular values, for any particular galaxy, that are not known.

2.1 A simple linear model

This section develops statistical methods for a simple linear model of the form (1). This allows the key concepts of linear modelling to be reviewed without the distraction of any mathematical difficulty.

Formally, consider n pairs of observations, (x_i, y_i), where y_i is an observation on random variable, Y_i, with expectation, $\mu_i \equiv \mathbb{E}(Y_i)$. Suppose that an appropriate model for the relationship between x and y is:

$$Y_i = \mu_i + \epsilon_i, \text{ where } \mu_i = x_i\beta. \tag{2}$$

Here β is an unknown parameter and the ϵ_i are mutually independent zero mean random variables, each with the same variance σ^2. So the model says that Y is given by x multiplied by a constant plus a random term. Y is an example of a *response (or outcome or dependent) variable*, while x is an example of a *predictor (or covariate or independent) variable*. Figure 2 illustrates this model for a case where $n = 8$.

In this regression model we are not including an intercept term, which is the point at which the regression line crosses the y-axis. In this case, because we have no intercept the regression line crosses the y-axis at the origin. And this makes sense, as when the distance is zero, the corresponding recessional velocity is also zero. In general, by default, linear regression models should include an intercept, unless theory or prior knowledge suggests that the relationship between the two variables goes through the origin. Omitting the intercept when it should be included can lead to biased estimates of the regression coefficients, leading to incorrect conclusions.

2.1.1 Simple least squares estimation

How can β, in model (2), be estimated from the $(x_i, y_i)_{i=1}^n$ data? A sensible approach is to choose a value of β that makes the model fit closely to the data. To do this we need to define a measure of how well, or how badly, a model with a particular β fits the data. One possible measure is the **residual sum of squares** (RSS) of the model:

$$\mathcal{S} = \sum_{i=1}^{n} (y_i - \mu_i)^2.$$

For the current model the RSS is

$$\mathcal{S} = \sum_{i=1}^{n} (y_i - x_i\beta)^2.$$

If we have chosen a good value of β, close to the true value, then the model predicted μ_i should be relatively close to the y_i, so that $\mathcal{S}$ should be small, whereas poor choices will lead to μ_i far from their corresponding y_i, and high values of $\mathcal{S}$. Hence β can be estimated by minimizing $\mathcal{S}$ with respect to (w.r.t.) β and this is known as the method of *least squares*.

To minimize $\mathcal{S}$, for the current simple model, differentiate w.r.t. β:

$$\frac{\partial \mathcal{S}}{\partial \beta} = -\sum_{i=1}^{n} 2x_i(y_i - x_i\beta)$$

and set the result to zero to find $\hat{\beta}$, the least squares estimate of β:

$$-\sum_{i=1}^{n} 2x_i(y_i - x_i\hat{\beta}) = 0 \Rightarrow \sum_{i=1}^{n} x_i y_i - \hat{\beta}\sum_{i=1}^{n} x_i^2 = 0 \Rightarrow \hat{\beta} = \sum_{i=1}^{n} x_i y_i / \sum_{i=1}^{n} x_i^2.^*$$

2.2 Sampling properties of $\hat{\beta}$

To evaluate the reliability of the least squares estimate, $\hat{\beta}$, it is useful to consider the sampling properties of $\hat{\beta}$. That is, we should consider some properties of the distribution of $\hat{\beta}$ values, which would be obtained from repeated independent replication of the (x_i, y_i) data used for estimation. To do this, it is helpful to introduce the concept of an *estimator*, which is obtained by replacing the observations, y_i, in the estimate of $\hat{\beta}$ by the random variables, Y_i, to obtain

$$\hat{\beta} = \sum_{i=1}^{n} x_i Y_i / \sum_{i=1}^{n} x_i^2.$$

Clearly the *estimator*, $\hat{\beta}$, is a random variable and we can therefore discuss its distribution. For now, consider only the first two moments of that distribution.

The expected value of $\hat{\beta}$ is obtained as follows:

$$\mathbb{E}(\hat{\beta}) = \mathbb{E}\left(\sum_{i=1}^{n} x_i Y_i / \sum_{i=1}^{n} x_i^2\right) = \sum_{i=1}^{n} x_i \mathbb{E}(Y_i) / \sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} x_i^2 \beta / \sum_{i=1}^{n} x_i^2 = \beta.$$

So $\hat{\beta}$ is an unbiased estimator — its expected value is equal to the true value of the parameter that it is supposed to estimate.

Unbiasedness is a reassuring property, but knowing that an estimator gets it right on average, does not tell us much about how good any one particular estimate is likely to be: for this we also need to know how much estimates would vary from one replicate data set to the next — we need to know the estimator variance.

From general probability theory we know that if $Y_1, Y_2, \ldots, Y_n$ are *independent* random variables and $a_1, a_2, \ldots a_n$ are real constants then

$$\mathrm{var}\left(\sum_i a_i Y_i\right) = \sum_i a_i^2 \mathrm{var}(Y_i).$$

But we can write

$$\hat{\beta} = \sum_i a_i Y_i \ \text{ where } \ a_i = x_i / \sum_i x_i^2,$$

and from the original model specification we have that $\mathrm{var}(Y_i) = \sigma^2$ for all i. Hence,

$$\mathrm{var}(\hat{\beta}) = \frac{\sum_i x_i^2 \mathrm{var}(Y_i)}{\left(\sum_i x_i^2\right)^2} = \frac{\sigma^2}{\sum_i x_i^2}. \tag{3}$$

$^*\partial^2\mathcal{S}/\partial\beta^2 = 2\sum x_i^2$ which is clearly positive, so a minimum of $\mathcal{S}$ has been found.

In most circumstances σ^2 itself is an unknown parameter and must also be estimated. Since σ^2 is the variance of the ϵ_i, it makes sense to estimate it using the variance of the 'estimated' ϵ_i, the model **residuals**. In general these are defined as $\hat{\epsilon}_i = y_i - \hat{\mu}_i$, where $\hat{\mu}_i$ are the **fitted values**, the model estimates of $\mu_i = \mathbb{E}(y_i)$. For the current simple model $\hat{\mu}_i = x_i\hat{\beta}$, so $\hat{\epsilon}_i = y_i - x_i\hat{\beta}$. An unbiased estimator of σ^2 for this model is then:

$$\hat{\sigma^2} = \frac{1}{n-1}\sum_i (y_i - x_i\hat{\beta})^2 \tag{4}$$

(proof of unbiasedness is given later for the general case). Plugging this estimate of σ^2 into (3) obviously gives an unbiased estimate of the variance of $\hat{\beta}$.

2.3 So how old is the universe?

The least squares calculations derived above are available as part of the statistical package and environment R. The function `lm` fits linear models to data, including the simple example currently under consideration. The Cepheid distance — velocity data shown in figure 1 are stored in a data frame[†] The following R code fits the model and produces the output shown (and more!).

```
library(gamair); data(hubble)
#To know more about the data
help(hubble)

#Fit simple linear regression model (with no intercept)
hub.mod <- lm(y ~ x - 1, data = hubble)
summary(hub.mod)$coefficients

##   Estimate Std. Error t value     Pr(>|t|)
## x 76.58117   3.964794 19.3153 1.031907e-15
```

The call to `lm` passed two arguments to the function. The first is a *model formula*, `y~x-1`, specifying the model to be fitted: the name of the response variable is to the left of '~' while the predictor variable is specified on the right; the '`-1`' term indicates that the model has no 'intercept' term, i.e. that the model is a straight line through the origin. The second (optional) argument gives the name of the data frame in which the variables are to be found. `lm` takes this information and uses it to fit the model by least squares: the results are returned in a 'fitted model object', which in this case has been assigned to an object called `hub.mod` for later examination. '`<-`' is the assignment operator, and `hub.mod` is created by this assignment (overwriting any previously existing object of this name).

The `summary` function is then used to examine the fitted model object. Only part of its output is shown here and relevant to us so far: $\hat{\beta}$ and the estimate of the standard error of $\hat{\beta}$ (the square root of the estimated variance of $\hat{\beta}$, derived above). The column `t value`, is not relevant for now, but it gives the value of the test statistic associated to testing $H_0 : \beta = 0$ versus $H_1 : \beta \neq 0$. The last column gives the p-value associated with this test. Before using these quantities it is important to check the model assumptions. In particular we should check the plausibility of the assumptions that the ϵ_i are independent and all have the same variance. The way to do this is to examine residual plots.

Recall that the 'fitted values' for this model are defined as $\hat{\mu}_i = \hat{\beta}x_i$, while the residuals are simply $\hat{\epsilon}_i = y_i - \hat{\mu}_i$. A plot of residuals against fitted values is often useful and the following produces the plot in figure 3(a).

```
plot(fitted(hub.mod), residuals(hub.mod),
     xlab="fitted values", ylab="residuals")
```

What we would like to see, in such a plot, is an apparently random scatter of residuals around zero, with no trend in either the mean of the residuals, or their variability, as the fitted values increase. A trend in the mean violates the independence assumption, and is usually indicative of something missing in the model structure, while

[†] A data frame is just a two dimensional array of data in which the values of different variables are stored in different named columns.

.

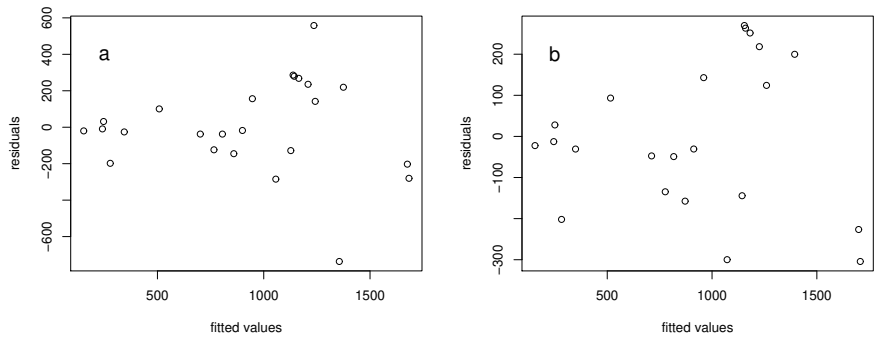


Figure 3: Residuals against fitted values for (a) the model (1) fitted to all the data in figure 1 and (b) the same model fitted to data with two substantial outliers omitted. Note the difference in the y-axis scales.

a trend in the variability violates the constant variance assumption. The main problematic feature of figure 3(a) is the presence of two points with very large magnitude residuals, suggesting a problem with the constant variance assumption. It is probably prudent to repeat the model fit, with and without these points, to check that they are not having undue influence on our conclusions. The following code omits the offending points and produces a new residual plot shown in figure 3(b).

```
#Extracting the index of the outlying observations
as.numeric(which(residuals(hub.mod) > 400 | residuals(hub.mod) < -600))

## [1]  3 15

#Fitting model without outliers
hub.mod1 <- lm(y ~ x - 1, data = hubble[-c(3, 15), ])
#Another way of extracting the coefficients from the summary output
coef(summary(hub.mod1))

##   Estimate Std. Error  t value     Pr(>|t|)
## x 77.67292    2.97028 26.15003 1.659996e-17

plot(fitted(hub.mod1), residuals(hub.mod1),
     xlab="fitted values", ylab="residuals")
```

The omission of the two large outliers has improved the residuals and changed $\hat{\beta}$ somewhat, but not drastically [‡]. The Hubble constant estimates have units of $(\mathrm{km})\mathrm{s}^{-1}\ (\mathrm{Mpc})^{-1}$. A Mega-parsec is $3.09 \times 10^{19}\mathrm{km}$, so we need to divide $\hat{\beta}$ by this amount, in order to obtain Hubble's constant with units of s^{-1}. The approximate age of the universe, in seconds, is then given by the reciprocal of $\hat{\beta}$. Here are the two possible estimates expressed in years:

```
hubble.const <- c(coef(hub.mod), coef(hub.mod1))/3.09e19
#Approximate age of the universe in seconds
age <- 1/hubble.const
#Approximate age of the universe in years
age/(60^2*24*365)
```

[‡]Note that in general it is not good practice to remove the outlying observations unless there is a strong motivation to do (e.g., a negative glucose level or cholsterol level of say 1000) and we should make this choice clear in our analysis (and report the results with and without the outlying observations). Robust regression methods, which we will not cover, assign low weight to outlying observations so that they do not impact (so much) the corresponding least squares estimates.

```
##           x           x
## 12794692825 12614854757
```

Both fits give an age of around 13 billion years. So we now have an idea of the best estimate of the age of the universe, but what range of ages would be consistent with the data?

2.4 Adding a distributional assumption

So far everything done with the simple model has been based only on the model equations and the two assumptions of independence and equal variance, for the response variable. If we wish to go further, and find confidence intervals for β, or test hypotheses related to the model, then a further distributional assumption will be necessary.

Specifically, assume that $\epsilon_i \sim N(0, \sigma^2)$ for all i, which is equivalent to assuming $Y_i \sim N(x_i\beta, \sigma^2)$. Now we have already seen that $\hat{\beta}$ is just a weighted sum of Y_i, but the Y_i are now assumed to be normal random variables, and a weighted sum of normal random variables is itself a normal random variable. Hence the estimator, $\hat{\beta}$, must be a normal random variable. Since we have already established the mean and variance of $\hat{\beta}$, we have that

$$\hat{\beta} \sim N\left(\beta, \left(\sum x_i^2\right)^{-1} \sigma^2\right). \tag{5}$$

2.4.1 Testing hypotheses about β

One thing we might want to do is to try and evaluate the consistency of some hypothesized value of β with the data. For example some Creation Scientists estimate the age of the universe to be 6000 years, based on a reading of the Bible. This would imply that $\beta = 163 \times 10^6$[§]. The consistency with data of such a hypothesized value for β can be based on the probability that we would have observed the $\hat{\beta}$ actually obtained, if the true value of β was the hypothetical one.

Specifically, we can test the null hypothesis, $\mathrm{H}_0 : \beta = \beta_0$, versus the alternative hypothesis, $\mathrm{H}_1 : \beta \neq \beta_0$, for some specified value β_0, by examining the probability of getting the observed $\hat{\beta}$, or one further from β_0, assuming H_0 to be true. If σ^2, the variance of the error term, were known then we could work directly from (5), as follows.

The probability required is known as the **p-value** of the test. It is

> *the probability of getting a value of $\hat{\beta}$ at least as favourable to H_1 as the one actually observed, if H_0 is actually true*.

Actually this definition holds for the p-value of any hypothesis test, if the specific '$\hat{\beta}$' is replaced by the general '*a test statistic*'. In this case it helps to distinguish notationally between the estimate, $\hat{\beta}_{\mathrm{obs}}$, and estimator $\hat{\beta}$. The p-value is then

$$\begin{aligned} p &= \Pr\left[|\hat{\beta} - \beta_0| \geq |\hat{\beta}_{\mathrm{obs}} - \beta_0| \middle| \mathrm{H}_0\right] \\ &= \left[\frac{|\hat{\beta} - \beta_0|}{\sigma_{\hat{\beta}}} \geq \frac{|\hat{\beta}_{\mathrm{obs}} - \beta_0|}{\sigma_{\hat{\beta}}} \middle| \mathrm{H}_0\right] \\ &= \Pr[|Z| > |z|] \end{aligned}$$

where $Z \sim N(0,1)$, $z = (\hat{\beta}_{\mathrm{obs}} - \beta_0)/\sigma_{\hat{\beta}}$ and $\sigma^2_{\hat{\beta}} = (\sum x_i^2)^{-1}\sigma^2$. Here Z is the test statistic (a random variable) and z is the realisation/observed value of the test statistic. Hence, having formed z, the p-value is easily worked out, using the cumulative distribution function for the standard normal. Small p-values suggest that the data are inconsistent with H_0, while large values indicate consistency. 0.05 is often used as the boundary between 'small' and 'large' in this context. In reality σ^2 is usually unknown. Broadly the same testing procedure can still be adopted, by replacing σ with $\hat{\sigma}$ (see Equation (4)), but we need to somehow allow for the extra uncertainty that this introduces (unless the sample size is very large). It turns out that if $\mathrm{H}_0 : \beta = \beta_0$ is true then

$$T \equiv \frac{\hat{\beta} - \beta_0}{\hat{\sigma}_{\hat{\beta}}} \sim t_{n-1}$$

[§]This isn't really valid, of course, since the Creation Scientists are not postulating a big bang theory.

where n is the sample size, $\hat{\sigma}^2_{\hat{\beta}} = (\sum x_i^2)^{-1}\hat{\sigma}^2$, and t_{n-1} is the t-distribution with $n-1$ degrees of freedom. The value $n-1$ comes from the number of data minus the number of estimated model parameters, and the result is proven in section 4.3.1. It is clear that large magnitude values of T favour H_1, so using T as the test statistic, in place of Z, we can calculate a p-value by evaluating

$$p = \Pr[|T| > |t|]$$

where $T \sim t_{n-1}$ and $t = (\hat{\beta}_{\text{obs}} - \beta_0)/\hat{\sigma}_{\hat{\beta}}$. Here is some code to evaluate the p-value for H_0 : *the Hubble constant is 163000000*.

```
cs.hubble <- 163000000
#test statistic
t.stat <- (coef(hub.mod1) - cs.hubble)/summary(hub.mod1)$coefficients[2]
t.stat

##          x
## -54876951

#An Alternative way to extract the regression coefficient
t.stat <- (summary(hub.mod1)$coefficients[1] - cs.hubble)/
  summary(hub.mod1)$coefficients[2]
t.stat

## [1] -54876951

#p-value
df <- nrow(hubble[-c(3, 15), ]) - 1
df

## [1] 21

pt(t.stat, df = df)*2 # 2 because of |T| in p-value defn.

## [1] 3.906388e-150

#Alternatively
pt(abs(t.stat), df = df, lower = FALSE)*2

## [1] 3.906388e-150
```

As judged by the test statistic, t, the data would be hugely improbable if $\beta = 1.63 \times 10^8$. It would seem that the hypothesized value can be rejected rather firmly (in this case, using the data with the outliers increases the p-value by a factor of 1000 or so).

Hypothesis testing is particularly useful when there are good reasons to want to stick with some null hypothesis, until there is good reason to reject it. This is often the case when comparing models of differing complexity: it is often a good idea to retain the simpler model until there is quite firm evidence that it is inadequate. Note one interesting property of hypothesis testing. If we choose to reject a null hypothesis whenever the p-value is less than some fixed level, α (often termed the *significance level* of a test), then we will inevitably reject a proportion, α, of correct null hypotheses. We could try and reduce the probability of such mistakes by making α very small, but in that case we pay the price of reducing the probability of rejecting H_0 when it is false!

2.4.2 Confidence intervals

Having seen how to test whether a *particular* hypothesized value of β is consistent with the data, the question naturally arises of what *range* of values of β would be consistent with the data? To answer this, we need to select a definition of 'consistent': a common choice is to say that any parameter value is consistent with the data if it results in a p-value of ≥ 0.05, when used as the null value in a hypothesis test.

Sticking with the Hubble constant example, and working at a significance level of 0.05, we would have rejected any hypothesized value for the constant, that resulted in a t value outside the range $(-2.08, 2.08)$, since these values would result in p-values of less than 0.05. The R function `qt` can be used to find such ranges: e.g. `qt(c(0.025,0.975),df=21)` returns the range of the middle 95% of t_{21} random variables. So we would accept any β_0 fulfilling:

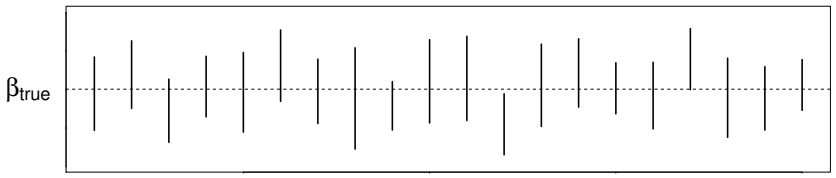
$$-2.08 \leq \frac{\hat{\beta} - \beta_0}{\hat{\sigma}_{\hat{\beta}}} \leq 2.08$$

which re-arranges to give the interval

$$\hat{\beta} - 2.08\hat{\sigma}_{\hat{\beta}} \leq \beta_0 \leq \hat{\beta} + 2.08\hat{\sigma}_{\hat{\beta}}.$$

Such an interval is known as a '95% Confidence interval' for β.

The defining property of a 95% confidence interval is this: if we were to gather an infinite sequence of independent replicate data sets, and calculate 95% confidence intervals for β from each, then 95% of these intervals would include the true β, and 5% would not. It is easy to see how this comes about. By construction, a hypothesis test with a significance level of 5% rejects the correct null hypothesis for 5% of replicate data sets, and accepts it for the other 95% of replicates. Hence 5% of 95% confidence intervals must exclude the true parameter, while 95% include it. The following illustrates this, showing intervals for a parameter β computed for 20 replicate datasets...



For the Hubble example, a 95% CI for the constant (in the usual astro-physicists units) is given by:

```
sigb <- summary(hub.mod1)$coefficients[2]
h.ci <- coef(hub.mod1) + qt(c(0.025, 0.975), df = df)*sigb
h.ci

## [1] 71.49588 83.84995
```

This can be converted to a confidence interval for the age of the universe, in years, as follows:

```
h.ci <- h.ci*60^2*24*365.25/3.09e19 # convert to 1/years
sort(1/h.ci)

## [1] 11677548698 13695361072
```

That s 95% CI is (11.7,13.7) billion years. Actually this 'Hubble age' is the age of the universe if it has been expanding freely, neither being slowed down by gravitation, nor accelerating for reasons not fully understood. In fact this free expansion assumption does not appear to be quite right, so our answer will be out by a bit more than the CI might suggest. But remember this is a very simple model, mainly for illustrative purposes.

3 Linear models in general

... The simple linear model, introduced above, can be generalized by allowing the response variable to depend on multiple predictor variables (plus an additive constant). These extra predictor variables can themselves be transformations of the original predictors. Here are some examples, for each of which a response variable datum, y_i, is treated as an observation on a random variable, Y_i, where $\mathbb{E}(Y_i) \equiv \mu_i$, the ϵ_i are zero mean random variables, and the β_j are model parameters, the values of which are unknown and will need to be estimated using data.

1. $\mu_i = \beta_0 + x_i\beta_1$, $Y_i = \mu_i + \epsilon_i$, is a straight line relationship between y and predictor variable, x.
2. $\mu_i = \beta_0 + x_i\beta_1 + x_i^2\beta_2 + x_i^3\beta_3$, $Y_i = \mu_i + \epsilon_i$, is a cubic model of the relationship between y and x.
3. $\mu_i = \beta_0 + x_i\beta_1 + z_i\beta_2 + \log(x_i z_i)\beta_3$, $Y_i = \mu_i + \epsilon_i$, is a model in which y depends on predictor variables x and z and on the log of their product.

Each of these is a linear model because the ϵ_i terms and the model parameters, β_j, enter the model in a linear way. Notice that the predictor variables can enter the model non-linearly. Exactly as for the simple model, the parameters of these models can be estimated by finding the β_j values which make the models best fit the observed data, in the sense of minimizing $\sum_i (y_i - \mu_i)^2$. The theory for doing this will be developed in section 4, and that development is based entirely on re-writing the linear model using matrices and vectors.

To see how this re-writing is done, consider the straight line model given above. Writing out the μ_i equations for all n pairs, (x_i, y_i), results in a large system of linear equations:

$$\begin{aligned} \mu_1 &= \beta_0 + x_1\beta_1 \\ \mu_2 &= \beta_0 + x_2\beta_1 \\ \mu_3 &= \beta_0 + x_3\beta_1 \\ . & \quad . \\ . & \quad . \\ \mu_n &= \beta_0 + x_n\beta_1 \end{aligned}$$

which can be re-written in matrix-vector form as

$$\begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ . \\ . \\ \mu_n \end{pmatrix} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \\ . & . \\ . & . \\ 1 & x_n \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix}.$$

So the model has the general form

$$\boldsymbol{\mu} = \mathbf{X}\boldsymbol{\beta}.$$

i.e. the expected value vector $\boldsymbol{\mu}$ is given by a **model matrix** (also known as a design matrix), $\mathbf{X}$, multiplied by a **parameter vector**, $\boldsymbol{\beta}$. All linear models can be written in this general form. Of course since $y_i = \mu_i + \epsilon_i$, we can also write the model as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon},$$

where $\mathbf{y} = (y_1, \ldots, y_n)'$ and $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)'$.

As a second illustration, the cubic example, given above, can be written in matrix vector form as

$$\begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ . \\ . \\ \mu_n \end{pmatrix} = \begin{pmatrix} 1 & x_1 & x_1^2 & x_1^3 \\ 1 & x_2 & x_2^2 & x_2^3 \\ 1 & x_3 & x_3^2 & x_3^3 \\ . & . & . & . \\ . & . & . & . \\ 1 & x_n & x_n^2 & x_n^3 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{pmatrix}.$$

Models in which data are divided into different groups, each of which are assumed to have a different mean, are less obviously of the form $\boldsymbol{\mu} = \mathbf{X}\boldsymbol{\beta}$, but in fact they can be written in this way, by use of dummy indicator variables. Again, this is most easily seen by example. Consider the model

$$\mu_i = \beta_j \text{ if observation } i \text{ is in group } j,$$

and suppose that there are three groups, each with 2 data. Then the model can be re-written

$$\begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \\ \mu_5 \\ \mu_6 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{pmatrix}.$$

Variables indicating the group to which a response observation belongs, are known as *factor* variables. Somewhat confusingly, the groups themselves are known as *levels* of a factor. So the above model involves one factor, 'group', with three levels. Models of this type, involving factors, are commonly used for the analysis of designed experiments. In this case the model matrix depends on the design of the experiment (i.e. on which units belong to which groups), and for this reason the terms 'design matrix' and 'model matrix' are often used interchangeably. Whatever it is called, $\mathbf{X}$ is absolutely central to understanding the theory of linear models, generalized linear models and generalized additive models.

3.1 Notation summary

- The *response variable* vector $\mathbf{y}$, has *expected value* vector $\boldsymbol{\mu} = \mathbb{E}(\mathbf{y})$.
- According to the linear model $\boldsymbol{\mu} = \mathbf{X}\boldsymbol{\beta}$, where $\mathbf{X}$ is the *model matrix* and $\boldsymbol{\beta}$ the *parameter vector* (or *coefficient vector*).
- Hence the linear model can also be written as $\mathbf{y} = \mathbf{X}\boldsymbol{\beta}+\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon}$ is a *random error* vector with independent components.
- $\mathbb{E}(\boldsymbol{\epsilon}) = \mathbf{0}$ and the covariance matrix of $\boldsymbol{\epsilon}$ is $\mathbf{V}_{\boldsymbol{\epsilon}} = \text{var}(\boldsymbol{\epsilon}) = \text{cov}(\boldsymbol{\epsilon}, \boldsymbol{\epsilon}) = \mathbb{E}[(\boldsymbol{\epsilon} - \mathbb{E}[\boldsymbol{\epsilon}])(\boldsymbol{\epsilon} - \mathbb{E}[\boldsymbol{\epsilon}])^{\mathsf{T}}] = \mathbb{E}(\boldsymbol{\epsilon}\boldsymbol{\epsilon}^{\mathsf{T}}) = \sigma^2\mathbf{I}_n$. This stems from the fact that ϵ_i is independent of ϵ_j and so $\text{cov}(\epsilon_i, \epsilon_j) = 0$, for all $i \neq j$ [¶]. The elements of the diagonal of the covariance matrix are given by $\text{cov}(\epsilon_i, \epsilon_i) = \text{var}(\epsilon_i) = \sigma^2$, for $i = 1, \ldots, n$.
- Once the model is estimated, the *parameter estimates* are denoted by $\hat{\boldsymbol{\beta}}$. This notation will also be used for the parameter *estimator*, the context indicating which is meant.
- The *fitted value* vector is $\hat{\boldsymbol{\mu}} = \mathbf{X}\hat{\boldsymbol{\beta}}$.
- The *residual* vector is $\hat{\boldsymbol{\epsilon}} = \mathbf{y} - \hat{\boldsymbol{\mu}}$.

3.2 R and linear models in general

R has functions for setting up model matrices automatically, on the basis of a *model formula* specifying the structure for the columns of the model matrix. We will now explore one example with more than one variable using the `cars` data frame, which contains data on stopping `distances` of cars against `speed` when the driver was first signalled to stop. It typically takes some fixed amount of 'reaction time', after the stop signal, for the driver to apply the brakes, hence the car will travel a distance proportional to initial speed before beginning to slow at all: so we need a linear dependence on `speed` in the model. The kinetic energy of a car is proportional to the square of its `speed`, and once the brakes are applied they can dissipate this energy (ultimately as heat) at a constant rate

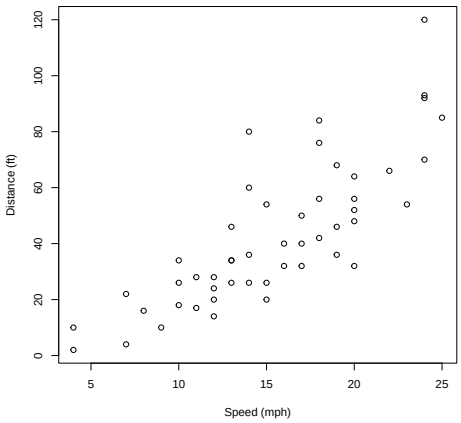
[¶] Recall that if two random variables are independent then their covariance is zero. However, the opposite is not necessarily true, i.e., two random variables can have a covariance of zero but still be dependent.

per unit distance travelled: hence the braking phase contributes a distance proportional to the square of `speed` to the total stopping distance. That is, a suitable model for the data is:

$$\texttt{distance}_i = \beta_0 + \beta_1 \texttt{speed}_i + \beta_2 \texttt{speed}_i^2 + \epsilon_i,$$

where the ϵ_i are i.i.d. $N(0, \sigma^2)$ random variables. Of course we would expect the parameter β_0 to be zero, given the justification for the model, but there is no harm in leaving it in for the moment. If there is statistical evidence that it is not zero, then this would suggest something wrong with our physical model: so it provides a useful check. Let us first look at the the data; to know more about the dataset we can type `help(cars)`.

```
data(cars)
# First rows of the dataset
head(cars)

##   speed dist
## 1     4    2
## 2     4   10
## 3     7    4
## 4     7   22
## 5     8   16
## 6     9   10

# Last rows of the dataset
tail(cars)

##    speed dist
## 45    23   54
## 46    24   70
## 47    24   92
## 48    24   93
## 49    24  120
## 50    25   85

# Structure of the data
str(cars)

## 'data.frame': 50 obs. of  2 variables:
##  $ speed: num  4 4 7 7 8 9 10 10 10 11 ...
##  $ dist : num  2 10 4 22 16 10 18 26 34 17 ...

# Summary of the data
summary(cars)

##      speed           dist
##  Min.   : 4.0   Min.   :  2.00
##  1st Qu.:12.0   1st Qu.: 26.00
##  Median :15.0   Median : 36.00
##  Mean   :15.4   Mean   : 42.98
##  3rd Qu.:19.0   3rd Qu.: 56.00
##  Max.   :25.0   Max.   :120.00

#Scatter plot of the data
plot(cars$speed, cars$dist,
     xlab = "Speed (mph)", ylab = "Distance (ft)")
```



The R function `model.matrix` can be used to set up the model matrix for this model of the `cars` data, as follows.

```
X <- model.matrix(~ speed + I(speed^2), data = cars)
```

Like `lm`, `model.matrix` takes a model formula and a data frame as arguments. It returns a model matrix. In this case the formula can be *one sided* as the response variable is immaterial when setting up the model matrix. Everything to the right of ~ in the formula specifies the structure of the model matrix. In this case we specify that there should be one column containing the `speed`s, one column containing the squares of the `speed`s and, since we did not suppress it, there will also be a column of `1`s, corresponding to the model intercept, β_0. Because some arithmetic symbols have special meanings (see later in notes) within model formulae, we have to 'protect' the term `speed^2`, by writing it as `I(speed^2)`, which means that we want `^2` to have its usual arithmetic meaning here. `I()` is the *identity function*: it just returns its argument. Here is what the first rows of `X` looks like...

```
head(X)
```

```
##   (Intercept) speed I(speed^2)
## 1           1     4         16
## 2           1     4         16
## 3           1     7         49
## 4           1     7         49
## 5           1     8         64
## 6           1     9         81
```

Having got R to set up an appropriate model matrix, we *could* use it much as we used the single covariate x in the Hubble data example...

```
stop.model0 <- lm(dist ~ X - 1, data = cars)
summary(stop.model0)
```

```
##
## Call:
## lm(formula = dist ~ X - 1, data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.720  -9.184  -3.188   4.628  45.152
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## X(Intercept)  2.47014   14.81716   0.167    0.868
## Xspeed        0.91329    2.03422   0.449    0.656
```

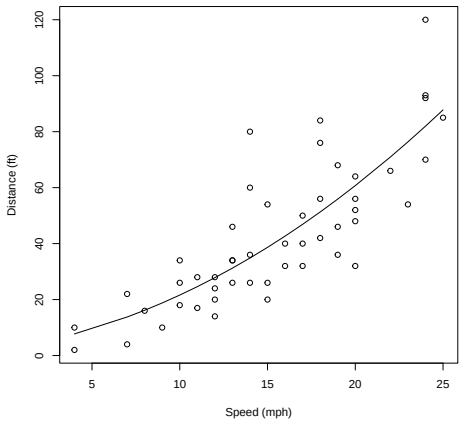
```
## XI(speed^2)   0.09996    0.06597   1.515    0.136
##
## Residual standard error: 15.18 on 47 degrees of freedom
## Multiple R-squared:  0.9133,Adjusted R-squared:  0.9078
## F-statistic: 165.1 on 3 and 47 DF,  p-value: < 2.2e-16
```

Notice the `Coefficients` section of the table, in particular. It now has a row for each model coefficient, with each row having much the same interpretation as we have already covered for the single parameter case. Actually, we would not usually form a model matrix explicitly, and then use `lm` in the way just done. Instead we would supply the more elaborate model formula directly to `lm`, from which it can create the appropriate model matrix automatically. For example:

```
stop.model <- lm(dist ~ speed + I(speed^2), data = cars)
summary(stop.model)

##
## Call:
## lm(formula = dist ~ speed + I(speed^2), data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.720  -9.184  -3.188   4.628  45.152
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.47014   14.81716   0.167    0.868
## speed        0.91329    2.03422   0.449    0.656
## I(speed^2)   0.09996    0.06597   1.515    0.136
##
## Residual standard error: 15.18 on 47 degrees of freedom
## Multiple R-squared:  0.6673,Adjusted R-squared:  0.6532
## F-statistic: 47.14 on 2 and 47 DF,  p-value: 5.852e-12
```

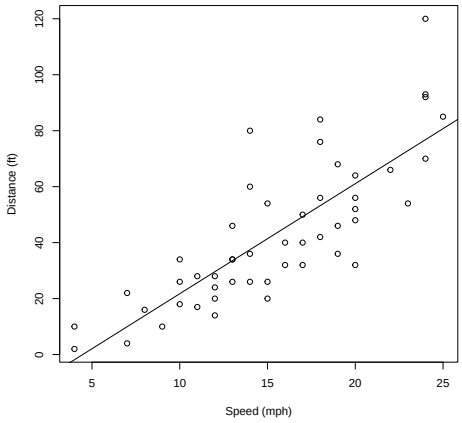
The results are, of course, identical to those produced by the more long winded approach. Well, we see that the R^2 and F statistic values in the summary output of the two models are different. The correct values are those obtained from `stop.model`; this is due to a silly thing that `R` does behind the scenes (this will be explained carefully in section 5). For now, we will proceed. Let us add the fitted values to the scatter plot of the data.

```
plot(cars$speed, cars$dist,
       xlab = "Speed (mph)", ylab = "Distance (ft)")

ss <- sort(cars$speed)
lines(ss, coef(stop.model)[1] + coef(stop.model)[2]*ss+ coef(stop.model)[3]*ss^2)
```



Out of curiosity, let us look at the fitted values for a model without the quadratic term.

```
stop.model2 <- lm(dist ~ speed, data = cars)
plot(cars$speed, cars$dist,
       xlab = "Speed (mph)", ylab = "Distance (ft)")
abline(stop.model2)
```



By looking at the two plots, the quadratic model seems to provide a better fit (later we will learn tools to perform model selection). Let us for a moment go back to the summary output of the quadratic model. Looking at the summary information in more detail, all the p-values for testing the single parameter hypotheses, $H_0 : \beta_j = 0$, are very high. Does this mean that the we cannot reject all such hypotheses, and conclude that all the parameter values could really be zero? Absolutely not! Each test is only valid if the other parameters are allowed to take non-zero values: this is because the parameter estimators are not independent and so the p-values can not be either. It is possible to test that all parameters except the intercept are simultaneously zero, and this is the test that the final p-value on the last line of the summary relates to: clearly such a hypothesis has no support from the data.

The high p-values in the summary do suggest that *something* could be removed from our model, and a sensible approach is to try removing the term with the highest p-value. This is the intercept, and on the physical grounds, given in the model motivation, we might expect the intercept to be zero. Actually, the confidence interval of the intercept is very wide.

```
coef(stop.model)[1] +
  qt(c(0.025,0.975), df = nrow(cars) - 3)*summary(stop.model)$coefficients[1,2]

## [1] -27.33815  32.27843
```

```r
#Alternatively we can use the function confint
confint(stop.model)[1,]
```

```
##     2.5 %    97.5 %
## -27.33815  32.27843
```

Let us then fit the model without an intercept.

```r
stop.model3 <- lm(dist ~ speed + I(speed^2) - 1, data = cars)
summary(stop.model3)
```

```
##
## Call:
## lm(formula = dist ~ speed + I(speed^2) - 1, data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.836  -9.071  -3.152   4.570  44.986
##
## Coefficients:
##            Estimate Std. Error t value Pr(>|t|)
## speed       1.23903    0.55997   2.213  0.03171 *
## I(speed^2)  0.09014    0.02939   3.067  0.00355 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.02 on 48 degrees of freedom
## Multiple R-squared:  0.9133,Adjusted R-squared:  0.9097
## F-statistic: 252.8 on 2 and 48 DF,  p-value: < 2.2e-16
```

Now both β_1 and β_2 have much lower associated p-values: there are no grounds for dropping any more model terms. We should, of course, check residual plots for this model, exactly as was done for the Hubble data, but for the moment we will move on.

4 Linear model theory

As in the simple one parameter case, point estimates of the linear model parameters, $\boldsymbol{\beta}$, can be obtained by the method of least squares; that is, by minimising the residual sum of squares

$$\mathcal{S}(\boldsymbol{\beta}) = \sum_{i=1}^{n} \epsilon_i^2 = \sum_{i=1}^{n} (y_i - \mu_i)^2, \quad \mu_i = x_i \boldsymbol{\beta},$$

where x_i is the ith row of the design matrix $\mathbf{X}$, which we recall has dimension $n \times p$. Fitting a model so that we minimise the squared error terms, or equivalently try to make the μ_i as close as possible to the y_i, has intuitive appeal. The following link leads to a set of notes with a *shiny app* that allows one to play around, in a simple linear model, with the residual sum of squares by changing the intercept and slope values.

https://bookdown.org/egarpor/PM-UC3M/lm-i-model.html

From now on the random vector, $\mathbf{y}$, and its observed value will not be distinguished notationally, the context making clear which is meant.

To use least squares with a linear model, written in general matrix-vector form, first recall the link between the Euclidean length of a vector and the sum of squares of its elements. If $\mathbf{v}$ is any vector of dimension, $n \times 1$, then $\|\mathbf{v}\|^2 \equiv \mathbf{v}^\mathsf{T}\mathbf{v} \equiv \sum_{i=1}^{n} v_i^2$. Hence

$$\begin{aligned}
\mathcal{S}(\boldsymbol{\beta}) = \|\mathbf{y} - \boldsymbol{\mu}\|^2 &= \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 \\
&= (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^\mathsf{T}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) \\
&= (\mathbf{y}^\mathsf{T} - \boldsymbol{\beta}^\mathsf{T}\mathbf{X}^\mathsf{T})(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) \\
&= \mathbf{y}^\mathsf{T}\mathbf{y} - \boldsymbol{\beta}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{y} - \mathbf{y}^\mathsf{T}\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\beta}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{X}\boldsymbol{\beta} \\
&= \mathbf{y}^\mathsf{T}\mathbf{y} - 2\mathbf{y}^\mathsf{T}\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\beta}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{X}\boldsymbol{\beta},
\end{aligned}$$

where from the penultimate to the last line we have exploited the fact that the terms $\boldsymbol{\beta}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{y}$ and $\mathbf{y}^\mathsf{T}\mathbf{X}\boldsymbol{\beta}$ (as well as $\mathbf{y}^\mathsf{T}\mathbf{y}$ and $\boldsymbol{\beta}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{X}\boldsymbol{\beta}$) are scalars. Thus, $\boldsymbol{\beta}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{y}$ is equal to its transpose $(\boldsymbol{\beta}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{y})^\mathsf{T}$ and we have that $\boldsymbol{\beta}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{y} = \mathbf{y}^\mathsf{T}\mathbf{X}\boldsymbol{\beta}$ and so, in particular,

$$-\boldsymbol{\beta}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{y} - \mathbf{y}^\mathsf{T}\mathbf{X}\boldsymbol{\beta} = -2\mathbf{y}^\mathsf{T}\mathbf{X}\boldsymbol{\beta}.$$

We minimise $\mathcal{S}(\boldsymbol{\beta})$ by setting the vector of first derivatives to zero and by showing that the matrix of second derivatives is positive definite. [‖] In order to do so, let us introduce the following three rules for the differentiation of vector functions. If **A** is a matrix and **a**, **u**, **v** are vectors of appropriate dimension so that all operations can be performed, then

1. $\frac{\partial \mathbf{v}^\mathsf{T}\mathbf{u}}{\partial \mathbf{u}} = \mathbf{v}$.
2. If A is symmetric, then
$$\frac{\partial \mathbf{u}^\mathsf{T}\mathbf{A}\mathbf{u}}{\partial \mathbf{u}} = 2\mathbf{A}\mathbf{u} = 2\mathbf{A}^\mathsf{T}\mathbf{u}.$$
3. $\frac{\partial \mathbf{A}\mathbf{u}}{\partial \mathbf{u}^\mathsf{T}} = \mathbf{A}$.

Now,

$$\begin{aligned}
\frac{\partial \mathcal{S}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} &= \frac{\partial(\mathbf{y}^\mathsf{T}\mathbf{y} - 2\mathbf{y}^\mathsf{T}\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\beta}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{X}\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} \\
&= 0 - 2\frac{\partial(\mathbf{y}^\mathsf{T}\mathbf{X}\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} + \frac{\partial(\boldsymbol{\beta}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{X}\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}
\end{aligned}$$

[‖] A symmetric matrix **A** is positive definite if the real number $\mathbf{v}^\mathsf{T}\mathbf{A}\mathbf{v} > 0$ for every nonzero real vector **v**.

Letting in item **1.** of the rule above $\mathbf{v} = \mathbf{X}^\mathsf{T}\mathbf{y}$ (so that $\mathbf{v}^\mathsf{T} = \mathbf{y}^\mathsf{T}\mathbf{X}$), we have that

$$\frac{\partial(\mathbf{y}^\mathsf{T}\mathbf{X}\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} = \mathbf{X}^\mathsf{T}\mathbf{y}.$$

In turn, letting in item **2.** of the differentiation rule $\mathbf{A} = \mathbf{X}^\mathsf{T}\mathbf{X}$, which is symmetric, we have that

$$\frac{\partial(\boldsymbol{\beta}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{X}\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} = 2\mathbf{X}^\mathsf{T}\mathbf{X}\boldsymbol{\beta}.$$

Therefore,

$$\frac{\partial\mathcal{S}(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} = 0 \Rightarrow -2\mathbf{X}^\mathsf{T}\mathbf{y} + 2\mathbf{X}^\mathsf{T}\mathbf{X}\boldsymbol{\beta} = 0 \Rightarrow \mathbf{X}^\mathsf{T}\mathbf{X}\boldsymbol{\beta} = \mathbf{X}^\mathsf{T}\mathbf{y}.$$

These are know as the *normal equations* and lead to (assuming that $\mathbf{X}$ is full column rank and therefore $\mathbf{X}^\mathsf{T}\mathbf{X}$ is invertible)

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}.$$

Let us confirm that $\hat{\boldsymbol{\beta}}$ is indeed a minimum. Taking second derivatives results in

$$\frac{\partial^2\mathcal{S}(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}\partial\boldsymbol{\beta}^\mathsf{T}} = \frac{\partial(-2\mathbf{X}^\mathsf{T}\mathbf{y} + 2\mathbf{X}^\mathsf{T}\mathbf{X}\boldsymbol{\beta})}{\partial\boldsymbol{\beta}^\mathsf{T}} = 2\mathbf{X}^\mathsf{T}\mathbf{X},$$

where in the last equality we have applied item **3.** from the differentiation rule with $\mathbf{A} = \mathbf{X}^\mathsf{T}\mathbf{X}$. If the columns of $\mathbf{X}$ are linearly independent, i.e., if the design matrix is full column rank ($\text{rank}(\mathbf{X}) = \mathbf{p}$), then the matrix $\mathbf{X}^\mathsf{T}\mathbf{X}$ is positive definite[**] and $\hat{\boldsymbol{\beta}}$ is a minimum.

It can be easily shown that $\widehat{\boldsymbol{\beta}}$ is unbiased (has expectation equal to the true $\boldsymbol{\beta}$):

$$\mathbb{E}(\hat{\boldsymbol{\beta}}) = \mathbb{E}[(\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}] = (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbb{E}(\mathbf{y}) = (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{X}\boldsymbol{\beta} = \boldsymbol{\beta}.$$

The covariance matrix of $\hat{\boldsymbol{\beta}}$ is also easy to derive. The covariance matrix of $\mathbf{y}$ is $\sigma^2\mathbf{I}_n$ since $\text{var}(\mathbf{y}) = \text{var}(\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}) = \text{var}(\boldsymbol{\epsilon}) = \sigma^2\mathbf{I_n}$. Then,[††]

$$\begin{aligned}
\mathbf{V}_{\hat{\boldsymbol{\beta}}} = \text{var}(\hat{\boldsymbol{\beta}}) &= \text{var}\{(\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}\} \\
&= (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\text{var}(\mathbf{y})((\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T})^\mathsf{T} \\
&= (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\sigma^2\mathbf{I}_n\mathbf{X}((\mathbf{X}^\mathsf{T}\mathbf{X})^\mathsf{T})^{-1} \\
&= \sigma^2(\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{X}(\mathbf{X}^\mathsf{T}\mathbf{X})^{-1} \\
&= \sigma^2(\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}
\end{aligned}$$

There is a 'rule' in numerical linear algebra that says we should avoid computing the inverse of a matrix if it can be avoided. One way to circumvent the computation of inverse matrices when calculating the least squares estimates of $\boldsymbol{\beta}$ is to use the so-called QR decomposition, a method which will be introduced in the next section. In fact, this is the method implemented by the `lm` function. As an example of why we should avoid matrix inversion, we simulate data from an extreme example.

```
# Number of observations
n <- 50
# One predictor only
x <- seq(1, 500, len = n)
# Design matrix (polynomial regression)
X <- cbind(1, x, x^2, x^3) #Thus p = 4
colnames(X) <- c("Intercept","x","x2","x3")
```

[**] **Theorem**: Let $\mathbf{B}$ be an $n \times p$ matrix. The matrix $\mathbf{B}^\mathsf{T}\mathbf{B}$ is then symmetric and nonnegative definite. If $\text{rank}(\mathbf{B}) = p$, then the matrix $\mathbf{B}^\mathsf{T}\mathbf{B}$ is positive definite.

[††] Remember that $var(\mathbf{A}\mathbf{v} + \mathbf{b}) = \mathbf{A}\text{var}(\mathbf{v})\mathbf{A}^\mathsf{T}$. Also $(\mathbf{A}^{-1})^\mathsf{T} = (\mathbf{A}^\mathsf{T})^{-1}$.

```r
# True parameter
beta <- matrix(c(1, 1, 1, 1), nrow = 4, ncol = 1)
# Fixing the seed so that results are reproducible
set.seed(1)
# Simulating the response data
y <- X%*%beta + rnorm(n)

# OLS estimate of beta
solve(t(X)%*%X)%*%t(X)%*%y
```

```
## Error in solve.default(t(X) %*% X): system is computationally singular:  reciprocal
condition number = 2.93617e-17
```

```r
#Equivalent way
solve(crossprod(X))%*%crossprod(X,y)
```

```
## Error in solve.default(crossprod(X)): system is computationally singular:  reciprocal
condition number = 2.93617e-17
```

Oh, we got an error! Let us use the `lm` function and see what happens, i.e., if it also returns an error or if it is able to compute the least squares estimate.

```r
lm_fit <- lm(y ~ X - 1)
lm_fit$coefficients
```

```
## XIntercept          Xx         Xx2         Xx3
##  0.9038373  1.0066440  0.9999622  1.0000001
```

We did not get an error and the estimated values are very close to the true values used to simulate the data. Let us then learn the method `lm` is using.

4.1 QR decomposition

We start by noting that since $\mathcal{S}$ is simply the squared (Euclidean) length of the vector $\mathbf{y} - \mathbf{X}\boldsymbol{\beta}$, its value will be unchanged if $\mathbf{y} - \mathbf{X}\boldsymbol{\beta}$ is rotated or reflected. This observation is the basis for the QR method for finding $\hat{\boldsymbol{\beta}}$ and for developing the distributional results required to use linear models.

Specifically, as with any real matrix, $\mathbf{X}$ can always be decomposed as

$$\mathbf{X} = \mathcal{Q}\begin{pmatrix} \mathbf{R} \\ \mathbf{0} \end{pmatrix} = \mathbf{Q}\mathbf{R}, \tag{6}$$

where $\mathbf{R}$ is a $p \times p$ upper triangular matrix[‡‡], $\mathbf{0}$ is a matrix of dimension $(n-p) \times p$ with all entries equal to zero, and $\mathcal{Q}$ is an $n \times n$ orthogonal matrix, the first p columns of which form $\mathbf{Q}$. Recall that an orthogonal matrix $\mathcal{Q}$ is a square matrix whose columns and rows are orthogonal unit vectors. It therefore holds that $\mathcal{Q}\mathcal{Q}^\mathsf{T} = \mathcal{Q}^\mathsf{T}\mathcal{Q} = \mathbf{I}_n$ or, equivalently, $\mathcal{Q}^\mathsf{T} = \mathcal{Q}^{-1}$. Also, orthogonal matrices rotate/reflect vectors, but do not change their length. Further note that $\mathbf{Q}$ is an $n \times p$ matrix with orthogonal columns, but non-orthogonal rows, so that $\mathbf{Q}^\mathsf{T}\mathbf{Q} = \mathbf{I}_p$ but $\mathbf{Q}\mathbf{Q}^\mathsf{T} \neq \mathbf{I}$. Multiplying $\mathbf{y} - \mathbf{X}\boldsymbol{\beta}$ by $\mathcal{Q}^\mathsf{T}$ implies that

$$\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 = \|\mathcal{Q}^\mathsf{T}\mathbf{y} - \mathcal{Q}^\mathsf{T}\mathbf{X}\boldsymbol{\beta}\|^2 = \left\|\mathcal{Q}^\mathsf{T}\mathbf{y} - \mathcal{Q}^\mathsf{T}\mathcal{Q}\begin{pmatrix} \mathbf{R} \\ \mathbf{0} \end{pmatrix}\boldsymbol{\beta}\right\|^2 = \left\|\mathcal{Q}^\mathsf{T}\mathbf{y} - \begin{pmatrix} \mathbf{R} \\ \mathbf{0} \end{pmatrix}\boldsymbol{\beta}\right\|^2.$$

Defining $\mathbf{f}$ as a p dimensional vector and $\mathbf{r}$ as a $n-p$ vector, so that $\begin{pmatrix} \mathbf{f} \\ \mathbf{r} \end{pmatrix} \equiv \mathcal{Q}^\mathsf{T}\mathbf{y}$, yields

$$\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 = \left\|\begin{pmatrix} \mathbf{f} \\ \mathbf{r} \end{pmatrix} - \begin{pmatrix} \mathbf{R} \\ \mathbf{0} \end{pmatrix}\boldsymbol{\beta}\right\|^2 = \left\|\begin{pmatrix} \mathbf{f} \\ \mathbf{r} \end{pmatrix} - \begin{pmatrix} \mathbf{R}\boldsymbol{\beta} \\ \mathbf{0} \end{pmatrix}\right\|^2 = \left\|\begin{pmatrix} \mathbf{f} - \mathbf{R}\boldsymbol{\beta} \\ \mathbf{r} \end{pmatrix}\right\|^2 = \|\mathbf{f} - \mathbf{R}\boldsymbol{\beta}\|^2 + \|\mathbf{r}\|^2.$$

[‡‡] That is, $R_{i,j} = 0$ if $i > j$

If the final equality is not obvious recall that $\|\mathbf{x}\|^2 = \sum_i x_i^2$, so if $\mathbf{x} = \begin{pmatrix} \mathbf{v} \\ \mathbf{w} \end{pmatrix}$, $\|\mathbf{x}\|^2 = \sum_i v_i^2 + \sum_i w_i^2 = \|\mathbf{v}\|^2 + \|\mathbf{w}\|^2$.

Recall that our goal is to find the vector $\boldsymbol{\beta}$ that minimises $\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2$ or, equivalently, the vector $\boldsymbol{\beta}$ that minimises $\|\mathbf{f} - \mathbf{R}\boldsymbol{\beta}\|^2 + \|\mathbf{r}\|^2$. The length of $\mathbf{r}$ does not depend on $\boldsymbol{\beta}$ and $\|\mathbf{f} - \mathbf{R}\boldsymbol{\beta}\|^2$ can be reduced to zero by choosing $\boldsymbol{\beta}$ so that $\mathbf{R}\boldsymbol{\beta}$ equals $\mathbf{f}$. Note that by construction, $\mathbf{f} = \mathbf{Q}^\mathsf{T}\mathbf{y}$ and so we have

$$\mathbf{R}\boldsymbol{\beta} = \mathbf{Q}^\mathsf{T}\mathbf{y}.$$

But, since $\mathbf{R}$ is an upper triangular matrix, this system of equations can be easily solved by back substitution, thus allowing to find $\hat{\boldsymbol{\beta}}$ without inverting matrices. Of course, provided that $\mathbf{X}$ and therefore $\mathbf{R}$ have full column rank, we have

$$\hat{\boldsymbol{\beta}} = \mathbf{R}^{-1}\mathbf{f}. \tag{7}$$

Further, inverting $\mathbf{R}$ is numerically more stable than inverting $\mathbf{X}^\mathsf{T}\mathbf{X}$. Notice that

$$\|\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}\|^2 = \|\mathbf{f} - \mathbf{R}\hat{\boldsymbol{\beta}}\|^2 + \|\mathbf{r}\|^2 = \|\mathbf{f} - \mathbf{R}\mathbf{R}^{-1}\mathbf{f}\|^2 + \|\mathbf{r}\|^2,$$

and thus $\|\mathbf{r}\|^2 = \|\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}\|^2$ is the *residual sum of squares* for the model fit.

Note that the estimate we have obtained before, $\hat{\boldsymbol{\beta}} = (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}$ is obviously equivalent to $\hat{\boldsymbol{\beta}} = \mathbf{R}^{-1}\mathbf{f}$

$$\begin{aligned}
\hat{\boldsymbol{\beta}} &= (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{y} \\
&= \left((\mathbf{Q}\mathbf{R})^\mathsf{T}\mathbf{Q}\mathbf{R}\right)^{-1}(\mathbf{Q}\mathbf{R})^\mathsf{T}\mathbf{y} \\
&= (\mathbf{R}^\mathsf{T}\mathbf{Q}^\mathsf{T}\mathbf{Q}\mathbf{R})^{-\mathbf{1}}(\mathbf{Q}\mathbf{R})^\mathsf{T}\mathbf{y} \\
&= (\mathbf{R}^\mathsf{T}\mathbf{R})^{-\mathbf{1}}(\mathbf{Q}\mathbf{R})^\mathsf{T}\mathbf{y} \\
&= \mathbf{R}^{-\mathbf{1}}(\mathbf{R}^\mathsf{T})^{-\mathbf{1}}\mathbf{R}^\mathsf{T}\mathbf{Q}^\mathsf{T}\mathbf{y} \\
&= \mathbf{R}^{-\mathbf{1}}\mathbf{Q}^\mathsf{T}\mathbf{y} \\
&= \mathbf{R}^{-1}\mathbf{f}
\end{aligned}$$

Of course, the same equivalence also holds for the covariance matrix. Note that the covariance matrix of $\mathbf{f} = \mathbf{Q}^\mathsf{T}\mathbf{y}$ is

$$\text{var}(\mathbf{f}) = \text{var}(\mathbf{Q}^\mathsf{T}\mathbf{y}) = \mathbf{Q}^\mathsf{T}\text{var}(\mathbf{y})\mathbf{Q} = \sigma^{\mathbf{2}}\mathbf{Q}^\mathsf{T}\mathbf{I}_\mathbf{n}\mathbf{Q} = \sigma^{\mathbf{2}}\mathbf{I}_\mathbf{p},$$

which implies that

$$\mathbf{V}_{\hat{\boldsymbol{\beta}}} = \text{var}(\hat{\boldsymbol{\beta}}) = \text{var}(\mathbf{R}^{-1}\mathbf{f}) = \mathbf{R}^{-1}\text{var}(\mathbf{f})\mathbf{R}^{-\mathsf{T}} = \mathbf{R}^{-\mathbf{1}}\mathbf{R}^{-\mathsf{T}}\sigma^{\mathbf{2}}.$$

Remember that we have found before that $\mathbf{V}_{\hat{\boldsymbol{\beta}}} = (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\sigma^2$. Finally, we only need to notice that

$$\begin{aligned}
\mathbf{V}_{\hat{\boldsymbol{\beta}}} &= (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\sigma^2 = \left(\mathbf{R}^\mathsf{T}\mathbf{Q}^\mathsf{T}\mathbf{Q}\mathbf{R}\right)^{-1}\sigma^2 = \left(\mathbf{R}^\mathsf{T}\mathbf{R}\right)^{-1}\sigma^2 \\
&= \mathbf{R}^{-1}\mathbf{R}^{-\mathsf{T}}\sigma^2.
\end{aligned}$$

A natural question that arises after all this is: how do we determine $\boldsymbol{\mathcal{Q}}$ and $\mathbf{R}$? Three methods for finding $\boldsymbol{\mathcal{Q}}$ and $\mathbf{R}$ are associated with the names of

- Householder: $\boldsymbol{\mathcal{Q}}$ is a product of reflections.
- Givens: $\boldsymbol{\mathcal{Q}}$ is a product of rotations.
- Gram-Schmidt: successive orthogonalization.

In practice we will be using the `lm` function which uses the QR decomposition (behind the scenes). There is also the `qr` function that implements the QR decomposition. Let us now get an 'hands on' understanding of the QR decomposition. To do so, we revisit the simulated data example which we used to illustrate how directly inverting $\mathbf{X}^\mathsf{T}\mathbf{X}$ can be problematic. In the code below `Q_orthogonal` denotes $\boldsymbol{\mathcal{Q}}$ and `Q` denotes the matrix $\mathbf{Q}$, which is formed by the first p rows of $\boldsymbol{\mathcal{Q}}$.

```r
# QR decomposition of matrix X
qrx <- qr(X)
# n times n orthogonal matrix Q
Q_orthogonal <- qr.Q(qrx, complete = TRUE)
# R matrix
R <- qr.R(qrx)
# Q matrix
Q <- qr.Q(qrx)
# Oh, R it is upper triangular (note that here p = 4)
R
```

```
##      Intercept         x        x2         x3
## [1,] -7.071068 -1771.302 -596425.3 -225914269
## [2,]  0.000000  1039.159  520618.6  235997829
## [3,]  0.000000     0.000 -136509.6 -102586942
## [4,]  0.000000     0.000       0.0   17591856
```

```r
# Let us look at properties of Q_orthogonal
dim(Q_orthogonal)
```

```
## [1] 50 50
```

```r
Qort_Qort_t <- Q_orthogonal%*%t(Q_orthogonal) #identity matrix
#alternatively the above could be computed as tcrossprod(Q_orthogonal, Q_orthogonal)

Qort_t_Qort <- t(Q_orthogonal)%*%Q_orthogonal  #identity matrix
#alternatively the above could be computed as crossprod(Q_orthogonal, Q_orthogonal)

# Let us simulate a vector v and verify that its length will not change
# when multiplied by Q_orthogonal
set.seed(1)
v <- runif(n)
norm_v <- as.numeric(t(v)%*%v)
Q_orth_v <- Q_orthogonal%*%v
norm_Q_orth_v <- as.numeric(t(Q_orth_v)%*%Q_orth_v)
norm_v; norm_Q_orth_v
```

```
## [1] 17.81435
## [1] 17.81435
```

```r
# Now check the properties of Q
Q_t_Q <- crossprod(Q, Q) #identity matrix
Q_t_Q
```

```
##               [,1]          [,2]          [,3]          [,4]
## [1,]  1.000000e+00 -4.971820e-18 -1.946935e-17 -9.519193e-18
## [2,] -4.971820e-18  1.000000e+00 -1.654885e-16 -2.786901e-17
## [3,] -1.946935e-17 -1.654885e-16  1.000000e+00 -2.564948e-17
## [4,] -9.519193e-18 -2.786901e-17 -2.564948e-17  1.000000e+00
```

```r
Q_Q_t <- tcrossprod(Q, Q) #not an identity matrix

# Determine beta hat through the system R \beta = Q_t y
betahat <- backsolve(R, crossprod(Q,y))
betahat
```

```
##           [,1]
## [1,] 0.9038372
## [2,] 1.0066440
## [3,] 0.9999622
## [4,] 1.0000001
```

```r
# Directly through the function qr.solve
qr.solve(X, y)
```

```
##                [,1]
## Intercept 0.9038373
## x         1.0066440
## x2        0.9999622
## x3        1.0000001
```

```r
# Compare with lm results
lm_fit$coefficients
```

```
## XIntercept         Xx        Xx2        Xx3
##  0.9038373  1.0066440  0.9999622  1.0000001
```

```r
# Let us obtain f and r as defined above in the text
f <- t(Q)%*%y
# Or alternatively
p <- 4
aux <- t(Q_orthogonal)%*%y
f_alt <- aux[1:p]
f; f_alt
```

```
##            [,1]
## [1,] -226512473
## [2,]  236519487
## [3,] -102723452
## [4,]   17591857
## [1] -226512473  236519487 -102723452   17591857
```

```r
r <- aux[(p + 1):n]

backsolve(R, f)
```

```
##           [,1]
## [1,] 0.9038372
## [2,] 1.0066440
## [3,] 0.9999622
## [4,] 1.0000001
```

```r
# Alternatively
solve(R,f) # help(solve)
```

```
##                [,1]
## Intercept 0.9038372
## x         1.0066440
## x2        0.9999622
## x3        1.0000001
```

```r
# check that the norm of vector r indeed is the residual sum of squares (rss)
r_norm <- as.numeric(t(r)%*%r)
rss <- sum((y - X%*%betahat)^2)
r_norm; rss
```

```
## [1] 32.83243
## [1] 32.83243
```

```r
sum(lm_fit$residuals^2)
```

```
## [1] 32.83243
```

4.2 Checking

Further inference, beyond simply estimating $\boldsymbol{\beta}$, will require that the assumptions of independence and constant variance of the ϵ_i hold. This means that we need some means for checking these assumptions, if our inference is to be sound. Formally we will also assume normality of the ϵ_i, but in reality there is a certain robustness to this assumption, as a result of the central limit theorem.

The most useful checks of model assumptions are often graphical, as these tend to indicate not just that an assumption is violated, but also *how* it is violated. This in turn may allow us to fix the problem. Once we have estimated a linear model, all the information about model fit is contained in the model *residuals*

$$\hat{\epsilon}_i = y_i - \hat{\mu}_i$$

(where $\hat{\boldsymbol{\mu}} = \mathbf{X}\hat{\boldsymbol{\beta}}$, of course). To check the model assumptions the $\hat{\epsilon}_i$ are usually plotted against the model predictor variables and the *fitted values*, $\hat{\mu}_i$. Systematic pattern in the mean of the residuals will violate the independence assumption, and often results from something wrong with how the dependence on a predictor variable has been specified. Systematic pattern in the variability of the residuals indicates violation of the constant variance assumption. This is most often manifested in the plot of $\hat{\epsilon}_i$ against $\hat{\mu}_i$. We will come back to this in more detail in the next section.

4.3 Further inference results: intervals and testing

Having covered estimation and checking, we can now consider how to answer the inferential questions:

1. What range of parameter values are consistent with the data?
2. Are some pre-specified values (or restrictions) for the parameters consistent with the data?

In the specific case of linear models, we want to find confidence intervals for the β_i and to *test hypotheses* about the β_i. For example, we often want to test the null hypothesis that one or several elements of $\boldsymbol{\beta}$ are zero, corresponding to terms being omitted from the linear model.

As we have already seen in the case of a simple linear model, confidence intervals and hypothesis tests are probabilistic concepts, so at this stage we need to turn our linear model into a fully probabilistic model, by specifying a full distribution for the ϵ_i. We will simply assume that independently $\epsilon_i \sim N(0, \sigma^2)$ or, equivalently, that $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma^2\mathbf{I}_n)$ which, in turn, implies that $\mathbf{y} \sim N(\mathbf{X}\boldsymbol{\beta}, \mathbf{I}\sigma^2)$, and because $\hat{\boldsymbol{\beta}}$ is a linear transformation of $\mathbf{y}$, it follows that $\hat{\boldsymbol{\beta}}$ is also normally distributed

$$\hat{\boldsymbol{\beta}} \sim N(\boldsymbol{\beta}, \mathbf{V}_{\hat{\boldsymbol{\beta}}}), \text{ where } \mathbf{V}_{\hat{\boldsymbol{\beta}}} = \mathbf{R}^{-1}\mathbf{R}^{-\mathsf{T}}\sigma^2. \tag{8}$$

This is not yet in the most directly usable form: it gives the distribution of $\hat{\boldsymbol{\beta}}$ in terms of the unknowns $\boldsymbol{\beta}$ and σ^2. As we saw in section 2.4.1, if we want to make inferences about β_i it is helpful if we can start from a *pivotal* statistic, i.e., a function of the data and unobservable parameters whose distribution does not depend on any unknown parameters.

4.3.1 $(\hat{\beta}_i - \beta_i)/\hat{\sigma}_{\hat{\beta}_i} \sim t_{n-p}$

First, consider the distribution of $\mathbf{Q}^{\mathsf{T}}\mathbf{y}$. Again, it is a linear transform of a normal random vector, so must also be a normal random vector, and its covariance matrix is

$$\mathbf{V}_{\mathbf{Q}^{\mathsf{T}}\mathbf{y}} = \text{var}(\mathbf{Q}^{\mathsf{T}}\mathbf{y}) = \mathbf{Q}^{\mathsf{T}}\mathbf{I}\mathbf{Q}\sigma^2 = \mathbf{I}\sigma^2.$$

Since, for the normal distribution only, zero covariance implies independence, the elements of $\mathbf{Q}^{\mathsf{T}}\mathbf{y}$ are mutually independent. Furthermore,

$$\mathbb{E}\begin{pmatrix} \mathbf{f} \\ \mathbf{r} \end{pmatrix} = \mathbb{E}(\mathbf{Q}^{\mathsf{T}}\mathbf{y}) = \mathbf{Q}^{\mathsf{T}}\mathbb{E}(\mathbf{y}) = \mathbf{Q}^{\mathsf{T}}\mathbf{X}\boldsymbol{\beta} = \mathbf{Q}^{\mathsf{T}}\mathbf{Q}\begin{pmatrix} \mathbf{R} \\ \mathbf{0} \end{pmatrix}\boldsymbol{\beta} = \begin{pmatrix} \mathbf{R} \\ \mathbf{0} \end{pmatrix}\boldsymbol{\beta}$$

$$\Rightarrow \mathbb{E}(\mathbf{f}) = \mathbf{R}\boldsymbol{\beta} \text{ and } \mathbb{E}(\mathbf{r}) = \mathbf{0}_{n-p}.$$

So we have that

$$\mathbf{f} \sim N(\mathbf{R}\boldsymbol{\beta}, \mathbf{I}_p\sigma^2) \text{ and } \mathbf{r} \sim N(\mathbf{0}, \mathbf{I}_{n-p}\sigma^2)$$

with both vectors independent of each other (as, by construction, they are the constituents of $\boldsymbol{\mathcal{Q}}^\mathsf{T}\mathbf{y}$ whose elements are mutually independent).

Now, because the $n-p$ elements of $\mathbf{r}$ are i.i.d. $N(0,\sigma^2)$ random variables and, as such, the random variable $\frac{r_i}{\sigma}$, for $i = 1, \ldots, n-p$, also follows a normal distribution but with variance equal to one, leads to

$$\frac{1}{\sigma^2}\|\mathbf{r}\|^2 = \frac{1}{\sigma^2}\sum_{i=1}^{n-p} r_i^2 = \sum_{i=1}^{n-p}\left(\frac{r_i}{\sigma}\right)^2 \sim \chi^2_{n-p}.$$

Recall that because $r_i/\sigma \sim N(0,1)$, then $(r_i/\sigma)^2 \sim \chi^2_1$ and because all r_is are independent of each other for $i = 1, \ldots, n-p$, then $\sum_{i=1}^{n-p} r_i^2/\sigma^2 \sim \chi^2_{n-p}$.

The mean of a χ^2_{n-p} random variable is $n-p$, so this result is sufficient (but not necessary) to imply that

$$\hat{\sigma}^2 = \|\mathbf{r}\|^2/(n-p) \tag{9}$$

is an unbiased estimator of σ^2. To see why $\hat{\sigma}^2$ is unbiased observe that

$$\mathbb{E}(\hat{\sigma}^2) = \mathbb{E}\left(\frac{\|\mathbf{r}\|^2}{n-p}\right) = \frac{1}{n-p}\mathbb{E}\left(\frac{1}{\sigma^2}\sigma^2\|\mathbf{r}\|^2\right) = \frac{\sigma^2}{n-p}\mathbb{E}\left(\frac{1}{\sigma^2}\|\mathbf{r}\|^2\right) = \frac{\sigma^2}{n-p}(n-p) = \sigma^2.$$

The independence of the elements of $\mathbf{r}$ and $\mathbf{f}$ also implies that $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}^2$ are independent. Recall that $\mathbf{f}$ and $\mathbf{r}$ are independent and since $\hat{\boldsymbol{\beta}}$ depends only on the former and $\hat{\sigma}^2$ only on the latter, then $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}^2$ are independent.

Now consider a single-parameter estimator, $\hat{\beta}_i$, with standard deviation, $\sigma_{\hat{\beta}_i}$, given by the square root of element (i,i) of $\mathbf{V}_{\hat{\beta}} = \mathbf{R}^{-1}\mathbf{R}^{-\mathsf{T}}\sigma^2$. An unbiased estimator of $\mathbf{V}_{\hat{\beta}}$ is

$$\hat{\mathbf{V}}_{\hat{\beta}} = \mathbf{R}^{-1}\mathbf{R}^{-\mathsf{T}}\hat{\sigma}^2 = \left(\frac{1}{\sigma^2}\sigma^2\right)\mathbf{R}^{-1}\mathbf{R}^{-\mathsf{T}}\hat{\sigma}^2 = \frac{1}{\sigma^2}(\mathbf{R}^{-1}\mathbf{R}^{-\mathsf{T}}\sigma^2)\hat{\sigma}^2 = \frac{1}{\sigma^2}\mathbf{V}_{\hat{\beta}}\hat{\sigma}^2,$$

where the last equality will be helpful to determine the pivotal statistic. An estimator of $\sigma_{\hat{\beta}_i}$ is therefore given by the square root of element (i,i) of $\hat{\mathbf{V}}_{\hat{\beta}}$ and it is clear that $\hat{\sigma}_{\hat{\beta}_i} = \sigma_{\hat{\beta}_i}\hat{\sigma}/\sigma$. Hence, the pivotal statistic is

$$T = \frac{\hat{\beta}_i - \beta_i}{\hat{\sigma}_{\hat{\beta}_i}} = \frac{\hat{\beta}_i - \beta_i}{\sigma_{\hat{\beta}_i}\hat{\sigma}/\sigma} = \frac{(\hat{\beta}_i - \beta_i)/\sigma_{\hat{\beta}_i}}{\sqrt{\frac{1}{\sigma^2}\hat{\sigma}^2}} = \frac{(\hat{\beta}_i - \beta_i)/\sigma_{\hat{\beta}_i}}{\sqrt{\frac{1}{\sigma^2}\|\mathbf{r}\|^2/(n-p)}} \sim \frac{N(0,1)}{\sqrt{\chi^2_{n-p}/(n-p)}} \sim t_{n-p}, \tag{10}$$

where the independence of $\hat{\beta}_i$ and $\hat{\sigma}^2$ has been used, along with the definition of a t random variable (in particular, note that if $Z_1 \sim N(0,1)$ and independently $Z_2 \sim \chi^2_n$, then $T = Z_1/\sqrt{Z_2/n} \sim t_n$). This result can be used for any β_i exactly as it was used for the single β in sections 2.4.1 and 2.4.2 (with the implied adjustment of degrees of freedom, of course).

4.3.2 Testing $H_0 : \mathbf{C}\boldsymbol{\beta} = \mathbf{d}$

Now consider a more general testing problem. Recall that factor variables serve to classify observations into different groups. In a linear model each factor variable will usually have several associated parameters - in principle, one for each group. If we want to test whether the factor variable should be in the model or not, we therefore need to test whether all the factor's parameters could simultaneously be zero. This is a special case of the general null hypothesis $H_0 : \mathbf{C}\boldsymbol{\beta} = \mathbf{d}$ (against $H_1 : \mathbf{C}\boldsymbol{\beta} \neq \mathbf{d}$). Note that approaching such a testing problem with a separate test for each parameter of the factor is a bad idea: how would you go about combining the different p-values, for example?

Suppose that $\mathbf{C}$ is a $q \times p$ matrix and $\mathbf{d}$ a q dimensional vector, where $q < p$. Under H_0 we have $\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d} \sim N(\mathbf{0}, \mathbf{C}\mathbf{V}_{\hat{\beta}}\mathbf{C}^\mathsf{T})$, from basic properties of the transformation of normal random vectors. In particular, if

$\hat{\boldsymbol{\beta}} \sim N(\boldsymbol{\beta}, \mathbf{V}_{\hat{\boldsymbol{\beta}}})$, then $\mathbf{C}\hat{\boldsymbol{\beta}} \sim N(\mathbf{C}\boldsymbol{\beta}, \mathbf{C}\mathbf{V}_{\hat{\boldsymbol{\beta}}}\mathbf{C}^\mathsf{T})$ and because under H_0 one has $\mathbf{C}\boldsymbol{\beta} = \mathbf{d}$, we finally have that $\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d} \sim N(\mathbf{0}, \mathbf{C}\mathbf{V}_{\hat{\beta}}\mathbf{C}^\mathsf{T})$. Any positive definite matrix can be factored into the crossproduct of a triangular matrix with itself: a *Cholesky decomposition*. Forming the Cholesky decomposition $\mathbf{L}^\mathsf{T}\mathbf{L} = \mathbf{C}\mathbf{V}_{\hat{\beta}}\mathbf{C}^\mathsf{T}$ (if the design matrix $\mathbf{X}$ is full rank, then $\mathbf{V}_{\hat{\beta}}$ is positive definite), it is then easy to show that, under H_0, $\mathbf{L}^{-\mathsf{T}}(\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d}) \sim N(\mathbf{0}, \mathbf{I})$, so,

$$(\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d})^\mathsf{T}(\mathbf{C}\mathbf{V}_{\hat{\beta}}\mathbf{C}^\mathsf{T})^{-1}(\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d}) = (\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d})^\mathsf{T}\mathbf{L}^{-1}\mathbf{L}^{-\mathsf{T}}(\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d}) \sim \sum_{i=1}^{q} N(0,1)^2 \sim \chi^2_q.$$

Note that $[\mathbf{L}^{-\mathsf{T}}(\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d})]^\mathsf{T} = (\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d})^\mathsf{T}\mathbf{L}^{-1}$. As in the previous section, plugging in $\hat{\mathbf{V}}_{\hat{\beta}} = \mathbf{V}_{\hat{\beta}}\hat{\sigma}^2/\sigma^2$ gives the computable test statistic and its distribution under H_0:

$$\begin{aligned}
F &= \frac{1}{q}(\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d})^\mathsf{T}(\mathbf{C}\hat{V}_{\hat{\beta}}\mathbf{C}^\mathsf{T})^{-1}(\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d}) \\
&= \frac{1}{q}(\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d})^\mathsf{T}(\mathbf{C}\mathbf{V}_{\hat{\beta}}(\hat{\sigma}^2/\sigma^2)\mathbf{C}^\mathsf{T})^{-1}(\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d}) \\
&= \frac{\sigma^2}{q\hat{\sigma}^2}(\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d})^\mathsf{T}(\mathbf{C}V_{\hat{\beta}}\mathbf{C}^\mathsf{T})^{-1}(\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d}) \\
&= \frac{(\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d})^\mathsf{T}(\mathbf{C}V_{\hat{\beta}}\mathbf{C}^\mathsf{T})^{-1}(\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d})/q}{\frac{1}{\sigma^2}\|\mathbf{r}\|^2/(n-p)} \\
&\sim \frac{\chi^2_q/q}{\chi^2_{n-p}/(n-p)} \sim F_{q,n-p},
\end{aligned} \tag{11}$$

where again the independence between $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}^2$ was used. This result can be used to compute a p-value for the test. Recall that if $Z_n \sim \chi^2_n$ and independently $Z_m \sim \chi^2_m$, then $F = \frac{Z_n/n}{Z_m/m} \sim F_{n,m}$, a F distribution with n and m degrees of freedom. The F distribution, defined as the ratio of two chi-squared distributions, is restricted to the positive real line. As with the other tests, we reject the null hypothesis if the observed value of the F statistic, our test statistic, lies in an 'extreme' region of the corresponding F distribution. This is a one-sided test meaning that we reject H_0 in favour of H_1 if the F statistic is larger than the 95% quantile of the corresponding $F_{q,n-p}$ distribution (if we use a significance level of 5%, of course).

Note that when the null hypothesis is that a single coefficient is zero, then (11) reduces to (10). In this case, $\mathbf{d}$ is a scalar, and C is a row vector of dimension p with one of the entries equal to one and all the other $p-1$ entries are zero. Apart from testing the significance of factor variables, this test for general linear hypotheses has other interesting test problem as special cases. For instance, consider the following hypothetical model

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{3i} + \epsilon_i,$$

and suppose further that we want to test $H_0 : \beta_2 = \beta_3$ (or, equivalently, $H_0 : \beta_2 - \beta_3 = 0$) against the alternative $H_1 : \beta_2 \neq \beta_3$. Then, in this case, $\mathbf{C} = (0, 0, 1, -1)$ and $\mathbf{d} = 0$. Alternatively, suppose that for the same model interest instead lies in testing $\beta_2 = \beta_3 = 0$. The question of interest corresponding to this hypothesis is: Can y be explained by fewer variables? The alternative hypothesis, H_1 is that, at least, one of the two coefficients is different than zero. In this case we have $\mathbf{d} = (0, 0)^\mathsf{T}$ and $\mathbf{C} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$, leading to

$$\mathbf{C}\boldsymbol{\beta} = \begin{pmatrix} \beta_2 \\ \beta_3 \end{pmatrix} = \mathbf{d} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

When our interest is in testing whether several elements of $\boldsymbol{\beta}$ could simultaneously be zero, then $\mathbf{d} = \mathbf{0}$ while $\mathbf{C}$ is made up of a selection of rows of the $p \times p$ identity matrix (as we have just illustrated in the example above). In that case let RSS_1 denote the residual sum of squares for the full model, and RSS_0 denote the residual sum of squares for the reduced model in which the H_0 is true. It can be shown that the above test statistic then reduces to

$$F = \frac{(\mathrm{RSS}_0 - \mathrm{RSS}_1)/q}{\mathrm{RSS}_1/(n-p)}.$$

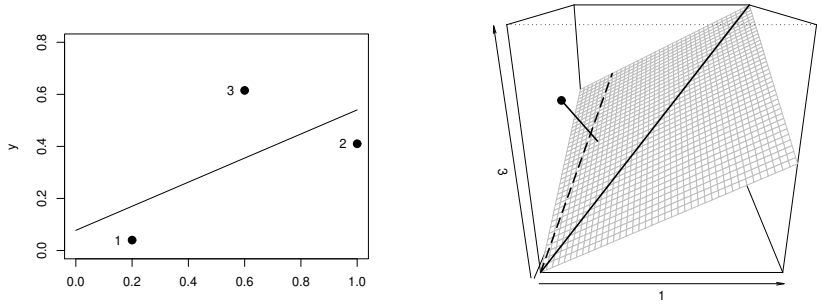


Figure 4: Illustration of the geometry of least squares. Left: a straight line fit to three (x, y) data points. Right: the space in which the y coordinates of the data define a single point, while the columns of the model matrix (solid and dashed line) span the subspace shown in grey. The least squares estimate of $\mathbb{E}(\mathbf{y})$ is the orthogonal projection of the data point onto the model subspace.

4.4 The geometry of linear models

Least squares estimation of linear models amounts to finding the orthogonal projection of the $n-$dimensional response vector $\mathbf{y}$ onto the p dimensional linear subspace spanned by the columns of $\mathbf{X}$. The linear model states that $\mathbb{E}(\mathbf{y}) = \boldsymbol{\mu} = \mathbf{X}\boldsymbol{\beta}$ lies in the space spanned by all possible linear combinations of the columns of the $n \times p$ model matrix $\mathbf{X}$, and least squares seeks the point in that space that is closest to $\mathbf{y}$ in Euclidean distance. That is, the least squares estimate of $\mathbb{E}(\mathbf{y})$ is the orthogonal projection of the data point in the space spanned by all possible linear combinations of the columns of $\mathbf{X}$. Figure 4 illustrates this geometry for the model,

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} .05 \\ .40 \\ .65 \end{pmatrix} = \begin{pmatrix} 1 & .1 \\ 1 & 1 \\ 1 & .6 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} + \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \end{pmatrix}.$$

The following youtube video provides a nice explanation of the geometric interpretation of the ordinary least squares

https://www.youtube.com/watch?v=oWuhZuLOEFY

Given this view of least squares fitting as orthogonal projection it is interesting to look at the projection matrix that maps the data $\mathbf{y}$ to the fitted values $\hat{\boldsymbol{\mu}}$. We have

$$\hat{\boldsymbol{\mu}} = \mathbf{X}\hat{\boldsymbol{\beta}} = \mathbf{Q}\mathbf{R}\mathbf{R}^{-1}\mathbf{Q}^{\mathsf{T}}\mathbf{y} = \mathbf{Q}\mathbf{Q}^{\mathsf{T}}\mathbf{y}.$$

So the required projection matrix is $\mathbf{A} = \mathbf{Q}\mathbf{Q}^{\mathsf{T}}$, which is often known as the *influence matrix*, or *hat matrix* of the linear model. In terms of the design matrix $A = \mathbf{X}(\mathbf{X}^{\mathsf{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathsf{T}}$. One interesting property of $\mathbf{A}$ is that it is idempotent: $\mathbf{A}\mathbf{A} = \mathbf{A}$. In a way this is obvious, as the orthogonal projection of $\hat{\boldsymbol{\mu}}$ onto the column space of $\mathbf{X}$ must be $\hat{\boldsymbol{\mu}}$. Also, $\mathbf{A} = \mathbf{A}^{\mathsf{T}}$. The hat matrix is important for checking assumptions of the linear model and for model diagnostics (more on this in the following section).

4.5 Maximum likelihood estimation of $\boldsymbol{\beta}$

We obtained the least squares estimator of $\boldsymbol{\beta}$ without any specific distributional assumptions regarding the error term $\boldsymbol{\epsilon}$. Assuming normally distributed errors, $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma^2\mathbf{I}_n)$, the maximum likelihood estimator for the unknown parameters can be computed and coincides with the least squares estimator. As we have seen, assuming normally distributed errors leads to $\mathbf{y} \sim N(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I})$, which yields the likelihood

$$L(\boldsymbol{\beta}, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left\{-\frac{1}{2\sigma^2}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^{\mathsf{T}}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})\right\}.$$

The log likelihood is thus given by

$$\ell(\boldsymbol{\beta}, \sigma^2) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\sigma^2) - \frac{1}{2\sigma^2}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^{\mathsf{T}}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}). \tag{12}$$

When maximising the log likelihood with respect to $\boldsymbol{\beta}$, we can ignore the first two terms in Equation (12) because they are independent of $\boldsymbol{\beta}$. Maximising $-\frac{1}{2\sigma^2}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^\mathsf{T}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})$ is equivalent to minimising $(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^\mathsf{T}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})$, which is the least squares criterion we have used to derive $\boldsymbol{\beta}$. The maximum likelihood estimator of $\boldsymbol{\beta}$ is therefore identical to the least squares estimator.

4.6 Further considerations

Before proceeding to the next section, a couple of considerations are worth.

- An effect β in a linear model with a single covariate, say x_1, is usually not the same as the effect, say β_1 of the same variable x_1 is a model with multiple covariates. The effect β is a *marginal* effect, ignoring all other potential covariates, whereas β_1 is a *conditional* effect, adjusting for the other covariates in the model. This should be obvious but always worth to reiterate.

- The theory of linear models presented so far has proceeded from the premise that design matrix $\mathbf{X}$ is fixed. If we repeat a study, we expect the response variable observations $\mathbf{y}$ to change, but if $\mathbf{X}$ is fixed, then the covariate values should be constant across replications of the study (*aside: how do you like this assumption for the Hubble's law example?*). This situation is realistically descriptive of an experiment, where the covariates are manipulated by the researcher. However, in observational studies (the standard, for instance, in social sciences and educational research) one would typically obtain different covariates on replication of the study and therefore $\mathbf{X}$ is random rather than fixed. Fortunately, the statistical theory of linear models, formulated under the supposition that the design matrix is fixed with respect to repeated sampling, is still valid, under mild conditions, when $\mathbf{X}$ is random.

5 Linear models in R

Having covered most of the theory needed to understand linear modelling, it is time to focus on application of that theory, so this section will cover the use of linear models in more detail.

5.1 The `cars` data again

We will use the `cars` data again in order to relate the theory covered in section 4 to practical analysis with R in the context of a now familiar example.

5.1.1 Confidence intervals and hypothesis tests for single parameters

This section will use the result of section 4.3.1, to find confidence intervals for, and test hypotheses about, model parameters. First refit the full quadratic model:

$$\texttt{dist}_i = \beta_1 + \beta_2 \texttt{speed}_i + \beta_3 \texttt{speed}_i^2 + \epsilon_i, \quad \epsilon_i \text{ i.i.d. } N(0, \sigma^2),$$

to the data in the `cars` data frame in R.

```
data(cars)
cm <- lm(dist ~ speed + I(speed^2), data = cars)
```

From the resulting fitted model object, `cm`, it is possible to extract all the quantities needed to construct the T statistics, covered in section 4.3.1. The following extracts, $\hat{\boldsymbol{\beta}}$, $\hat{\mathbf{V}}_{\hat{\boldsymbol{\beta}}}$ and the $\hat{\sigma}_{\hat{\beta}_i}$'s respectively:

```
beta.hat <- coef(cm)
# ALternative way of extracting the regression coefficients
# cm$coefficients
# summary(cm)$coefficients[,1]
V.beta <- vcov(cm)
sigma.beta <- sqrt(diag(V.beta))
```

Confidence intervals

Now, using the result from section 4.3.1, exactly as it was used in sections 2.4.1 and 2.4.2, we can find confidence intervals for the β_j's. Let's start by finding a 90% CI for β_2. This will involve using the `qt` function to evaluate the value above which 5% of t_{47} (note that $n = 50$ and p =3) distributed random variables lie (a further 5% lying below the negative of that value, by symmetry of the t distribution). For example

```
df <- nrow(cars) - 3
df

## [1] 47

qt(0.95, df = df)

## [1] 1.677927
```

implies that a t_{47} r.v. has a probability 0.05 of being greater than 1.678. The following returns the required confidence interval.

```
beta.hat[2] + qt(0.95, df = df)*sigma.beta[2]

##   speed
## 4.32656

beta.hat[2] - qt(0.95, df = df)*sigma.beta[2]

##     speed
## -2.499985
```

That is, a 90% CI for β_2 is $(-2.500, 4.327)$.
Actually, it is possible to calculate intervals for all three parameters at the same time. The following finds 95% CI's for all the parameters

```
beta.hat + qt(0.975, df = df)*sigma.beta

## (Intercept)       speed  I(speed^2)
##  32.2784284   5.0056113   0.2326702

beta.hat - qt(0.975, df = df)*sigma.beta

##  (Intercept)        speed   I(speed^2)
## -27.33815279  -3.17903606  -0.03275162
```

So, for example, a 95% CI for β_3 is $(-0.033, 0.233)$. Note that the `confint` function calculates these confidence intervals automatically.

```
confint(cm, level = 0.95)

##                     2.5 %     97.5 %
## (Intercept) -27.33815279 32.2784284
## speed        -3.17903606  5.0056113
## I(speed^2)   -0.03275162  0.2326702

# Also, for 90% CI for beta2
confint(cm, level = 0.9)[2, ]

##       5 %      95 %
## -2.499985  4.326560
```

Hypothesis tests

The result from section 4.3.1 is also useful for testing hypotheses about the values of the β_j. For example, consider testing $H_0 : \beta_2 = 0$ vs. $H_1 : \beta_2 \neq 0$. To test this we first form the t-statistic, $T = \hat{\beta}_2/\hat{\sigma}_{\hat{\beta}_2}$, and compute its observed value,

```
obs_t <- beta.hat[2]/sigma.beta[2]
obs_t

##    speed
## 0.448962
```

If H_0 is true then this should be an observation of a t_{47} random variable, but if H_1 is true then it should have too large a magnitude for this (basically because $\hat{\beta}_2$ will not then be 'close' to the hypothesized value, zero). As usual we quantify the evidence for or against H_0 using a p-value. This is given, in this case, by the following probability: $\Pr[|T| > |t|]$, where $T \sim t_{47}$ and $t = 0.449$. This is easily evaluated in R as follows.

```
p.val <- 2*pt(abs(obs_t), df = df, lower.tail = FALSE)
p.val

##     speed
## 0.6555224
```

A p-value of 0.66 suggests that the data are quite compatible with H_0. That is, provided that the model contains β_1 and β_3 terms, β_2 seems to be superfluous. However it might be that one of β_1 and β_3 is even more superfluous, so we should test whether either of those might be zero, before dropping any terms. In fact we can work out equivalent p-values for all 3 parameters simultaneously:

```
obs_t <- beta.hat/sigma.beta
p.val <- 2*pt(abs(obs_t), df = df,lower.tail=F)

#parameter estimates
beta.hat

## (Intercept)       speed  I(speed^2)
##   2.4701378   0.9132876   0.0999593

#estimated standard deviation of beta.hat
sigma.beta

## (Intercept)       speed  I(speed^2)
## 14.81716473  2.03422044  0.06596821

#observed t statistics
obs_t

## (Intercept)       speed  I(speed^2)
##   0.1667079   0.4489620   1.5152647

#p-values
p.val

## (Intercept)       speed  I(speed^2)
##   0.8683151   0.6555224   0.1364024
```

Actually, p-values for testing whether parameters might be zero are very often useful, so R produces them by default in the `summary` of a fitted linear model.

```
summary(cm)$coefficients

##              Estimate  Std. Error   t value  Pr(>|t|)
## (Intercept) 2.4701378 14.81716473 0.1667079 0.8683151
## speed       0.9132876  2.03422044 0.4489620 0.6555224
## I(speed^2)  0.0999593  0.06596821 1.5152647 0.1364024
```

Notice how the columns of the `Coefficients` table contain exactly the values that we have just calculated. These single parameter tests are often used to decide whether to drop a term from the model: usually the term with the highest p-value greater than some threshold, such as 0.05, would be dropped (provided that there are no prior grounds for insisting on its retention in the model). The model is refitted without the term, and the new p-values re-examined to see if any further simplification is possible. This is repeated until all terms have p-values lower than, say, 0.05. Note that the different estimates in $\hat{\boldsymbol{\beta}}$ are not independent (please find below $\hat{\mathbf{V}}_{\hat{\boldsymbol{\beta}}}$) and then dropping one term (setting it to zero) will change the estimates of the other coefficients and hence their p-values. This is why we need to remove one term at the time and refit the model (and not remove multiple parameters at a time, as the p-values were computed under the assumption that the other terms were in the model).

```
V.beta

##             (Intercept)       speed   I(speed^2)
## (Intercept) 219.5483705 -28.9523122  0.872858710
## speed       -28.9523122   4.1380528 -0.131439753
## I(speed^2)    0.8728587  -0.1314398  0.004351805
```

Different hypotheses and model selection

Sometimes other types of hypotheses may be tested. For example, it is hard to think of a decent physical interpretation for a negative β_1. Hence it might be more appropriate to test $\mathrm{H}_0 : \beta_1 = 0$ vs. $\mathrm{H}_1 : \beta_1 > 0$.

```
obs_t <- beta.hat[1]/sigma.beta[1]
obs_t

## (Intercept)
##   0.1667079
```

In this case large magnitude negative values of T count for H_0, so only the upper tail of the t_{47} distribution features in the p-value calculation: $\Pr[T > t]$ where $T \sim t_{47}$ and $t = 0.167$.

```
p.val <- pt(obs_t, df = df,lower.tail = F)
p.val

## (Intercept)
##   0.4341575
```

Again, no reason to reject the null (provided the other terms are in the model), but notice that the p-value is halved by the change in alternative.

Another use of the section 4.3.1 result is to test whether β_j might have some hypothesized value other than zero. For example we might be able to calculate what β_3 ought to be, based on the frictional properties of the brakes and tyres: suppose this calculation says that $\beta_3 = 0.2$. We can test this theoretical value, by testing $H_0 : \beta_3 = 0.2$ vs. $H_1 : \beta_3 \neq 0.2$, as follows.

```
obs_t <- (beta.hat[3] - 0.2)/sigma.beta[3]
p.val <- 2*pt(abs(obs_t), df = df, lower.tail = F)
p.val

## I(speed^2)
##   0.136091
```

So not much evidence with which to reject H_0. However, this test provides a very good example of why it is often a good idea to base statistical conclusions on the simplest model consistent with the data. If we repeat the test, based on a model without an intercept, then things are rather different.

```
cm1 <- lm(dist ~ speed + I(speed^2) - 1, data = cars)
beta.hat <- coef(cm1)
V.beta <- vcov(cm1)
sigma.beta <- sqrt(diag(V.beta))
obs_t <- (beta.hat[2]- 0.2)/sigma.beta[2]
p.val <- 2*pt(abs(obs_t), df = (nrow(cars) - 2), lower.tail = F)
p.val

##   I(speed^2)
## 0.0004934814
```

The p-value has dropped enormously, and we now have firm evidence to reject $H_0 : \beta_3 = 0.2$. What has happened here, is that the parameter estimators are so correlated in the full model, that all the estimators have very high variance. As a result we have a hard time rejecting a null hypothesis even if it quite a long way from the truth. The redundant parameters cause a loss of precision, and consequent loss of *power* (probability of correctly rejecting H_0) when testing. The more parsimonious model gives much better precision, and higher power when testing.

Note however, that there is a price to pay for *selecting* a model statistically and then performing hypothesis tests as if there were no question about the truth of the selected model. By doing this we fail to account for our initial uncertainty about which model was right when performing the hypothesis tests. This tends to mean that our p-values are a bit lower than if we had properly accounted for all uncertainty: i.e. the p-values are really only approximate now.

5.1.2 F-ratio tests about several parameters

This section illustrates the use of the F-ratio result from section 4.3.2, with R. We start by considering again the cars model with the following functional specification

$$\texttt{dist}_i = \beta_1 + \beta_2 \texttt{speed}_i + \beta_3 \texttt{speed}_i^2 + \epsilon_i, \quad \epsilon_i \text{ i.i.d. } N(0, \sigma^2).$$

Suppose that we had some prior reason to want to test:

$$\mathrm{H}_0 : \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \mathrm{H}_1 : \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix} \neq \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Note that H_1 states that, at least, one of the parameters is not zero. This is exactly the situation in which the results of section 4.3.2 are useful. Recall that the result is:

$$F = \frac{(\mathrm{RSS}_0 - \mathrm{RSS}_1)/(p_1 - p_0)}{\hat{\sigma}^2} \sim F_{p_1 - p_0, n - p_1} \text{ if } \mathrm{H}_0 \text{ is true.}$$

Here $p_1 - p_0$ is the difference in degrees of freedom between the models (difference in number of parameters) and$\hat{\sigma}^2 = \mathrm{RSS}_1/(n - p_1)$ is the residual variance estimator from the alternative (larger) model. Fitting the null model, it is then straightforward to calculate this F ratio statistic:

```
cm <- lm(dist ~ speed + I(speed^2), data = cars)
cm0 <- lm(dist ~ I(speed^2) - 1, data = cars)
rss0 <- sum(residuals(cm0)^2)
rss1 <- sum(residuals(cm)^2)
f_stat <- ((rss0 - rss1)/2)/(rss1/47)
f_stat

## [1] 2.412267
```

Note that, as claimed in section 4.3.2, the F test statistic in the way we have just computed it is equivalent to

$$F = \frac{1}{q}(\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d})^{\mathsf{T}}(\mathbf{C}\hat{V}_{\hat{\beta}}\mathbf{C}^{\mathsf{T}})^{-1}(\mathbf{C}\hat{\boldsymbol{\beta}} - \mathbf{d}),$$

where in this case $q = 2$ and

$$\mathbf{C} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \quad \text{and} \quad \mathbf{d} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

```
beta.hat <- coef(cm)
V.beta <- vcov(cm)
C <- matrix(c(1, 0, 0, 0, 1, 0), nrow = 2, ncol = 3, byrow = T)
d <- matrix(c(0, 0), ncol = 1, nrow = 2)
q <- 2
as.numeric(((t(C%*%beta.hat - d))%*%solve(C%*%V.beta%*%t(C))%*%(C%*%beta.hat - d))/q)

## [1] 2.412267
```

From the result of section 4.3.2 this ought to be an observation on an $F_{2,47}$ random variable, if H_0 is true. Otherwise it should be improbably large for an $F_{2,47}$ random variable. As usual we quantify the compatibility of the test statistic (and hence the data) with the stated distribution by calculating a p-value, which in this case is given by $\Pr[F > f]$ where $F \sim F_{2,47}$ and $f = 2.412$, which is easily evaluated using the `pf` function.

```
p.val <- pf(f_stat, 2, 47, lower.tail = F)
p.val

## [1] 0.1006278
```

The data provide little evidence to reject H_0, in this case.

While the above calculations are useful for linking the theory in section 4.3.2 to what is actually done to use the result for a model and data, there is usually no need to 'spell out' the calculations like this. R has built in functions to automate the process. For example the test just conducted can be done using the `anova` function as follows.

```
anova(cm0, cm)

## Analysis of Variance Table
##
## Model 1: dist ~ I(speed^2) - 1
## Model 2: dist ~ speed + I(speed^2)
##   Res.Df   RSS Df Sum of Sq      F Pr(>F)
## 1     49 11936
## 2     47 10825  2    1111.2 2.4123 0.1006
```

The quantities in this *ANOVA table* exactly match the quantities used in the previous, long winded, calculation.

Before proceeding it is worth mentioning that the last line in the `summary` object gives an F-statistic where the null hypothesis is that all coefficients, except the intercept, are zero. In this case $H_0 : \beta_2 = \beta_3 = 0$.

```
summary(cm)

##
## Call:
## lm(formula = dist ~ speed + I(speed^2), data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.720  -9.184  -3.188   4.628  45.152
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.47014   14.81716   0.167    0.868
## speed        0.91329    2.03422   0.449    0.656
## I(speed^2)   0.09996    0.06597   1.515    0.136
##
## Residual standard error: 15.18 on 47 degrees of freedom
## Multiple R-squared:  0.6673,Adjusted R-squared:  0.6532
## F-statistic: 47.14 on 2 and 47 DF,  p-value: 5.852e-12
```

It is also possible to compare a longer sequence of models using `anova`. Each pair in the sequence is compared in exactly the same way as two models would be compared, except that the residual variance estimate in the denominator of each F-ratio statistic is always taken from the 'largest' model (i.e. the one with most parameters).

Finally, the following is possible:

```
anova(cm)

## Analysis of Variance Table
##
## Response: dist
##            Df  Sum Sq Mean Sq F value    Pr(>F)
## speed       1 21185.5 21185.5  91.986 1.211e-12 ***
## I(speed^2)  1   528.8   528.8   2.296    0.1364
## Residuals  47 10824.7   230.3
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This single argument call causes `anova` to successively remove terms from your model, at each stage testing the null hypothesis that the current version of the model is correct, against the alternative that the previous version was

right. F-ratio tests are used each time, but again $\hat{\sigma}^2$ according to the original, larger, model is always used as the denominator of the F ratio statistic. The tables have to be read from bottom row up. So, the `I(speed^2)` row compares models with and without the dependence on `speed`^{2}, and suggests that we could drop it. Turning to the `speed` row, this is about comparing a model with intercept and linear dependence on `speed` to a model with only an intercept: it suggests that we should hang on to the `speed` term.

Actually this single argument use of `anova` only really makes sense in very specific situations. As we have already seen, if we reduce the `cars` model in a sensible order, we would drop the intercept term, and then nothing else.

5.1.3 Checking the model assumptions: residuals

In the previous few sections models for the `cars` data have been used in order to test hypotheses and find confidence intervals, but we have not actually checked that the model assumptions are reasonable. If the assumptions are not reasonable, then we have been wasting our time! For this reason it is usually important to check model assumptions immediately after fitting models, and before doing anything that relies on the assumptions being reasonable. The only reason that this was not done above, was in order to ensure that illustration of the practicalities of using the results of sections 4.3.1 and 4.3.2 followed as soon as possible after those sections.

The assumptions of the linear model are:

1. The structural part of the model, $\mathbb{E}(\mathbf{y}) = \mathbf{X}\boldsymbol{\beta}$, is correct. This assumption implies that the expectation includes all the important predictor variables in the precisely correct functional form and that it does not include any unimportant predictor variables.
2. The errors have constant variance (*homoscedasticity*).
3. The errors are independent.
4. The errors follow a normal distribution.

A couple of comments regarding two of these four assumptions are worth mentioning. With regard to assumption 1., as we have already learned in section 3, nonlinear effects of predictors can be accommodated in the framework of linear models as long as the resulting model is linear in the parameters. Regarding assumption 4., if the normality assumption does not hold, tests and confidence intervals are not exact. However, we can appeal to the central limit theorem which will ensure that the tests and confidence intervals constructed will be increasingly accurate approximations for larger sample sizes. Hence, we can afford to ignore the issue, provided the sample is sufficiently large or the violation not particularly severe (distributions that are not heavy tailed, not heavily multimodal, and not heavily skewed).

Although it is not part of regression diagnostics, it is worthwhile also mentioning that an even more important assumption is that the data at hand are relevant to the question of interest. Of course, this requires some qualitative judgement and domain knowledge and, by opposition to the four assumptions listed above, can not be checked through plots.

The errors are not observable but we can examine the residuals, $\hat{\boldsymbol{\epsilon}}$. Recall that the residuals, $\hat{\epsilon}_i$, are simply the response data, y_i, minus the fitted values, $\hat{\mu}_i$, i.e.

$$\hat{\boldsymbol{\epsilon}} = \mathbf{y} - \hat{\boldsymbol{\mu}}, \text{ where } \hat{\boldsymbol{\mu}} = \mathbf{X}\hat{\boldsymbol{\beta}}.$$

The residuals contain all the information left in the data after fitting the model. Note that

$$\mathbb{E}(\hat{\boldsymbol{\epsilon}}) = \mathbb{E}(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}) = \mathbb{E}(\mathbf{y}) - \mathbf{X}\mathbb{E}(\hat{\boldsymbol{\beta}}) = \mathbf{X}\boldsymbol{\beta} - \mathbf{X}\boldsymbol{\beta} = \mathbf{0}.$$

To evaluate the covariance matrix of $\hat{\boldsymbol{\epsilon}}$ we will make use of the influence matrix $\mathbf{A} = \mathbf{Q}\mathbf{Q}^{\mathsf{T}}$, which is of dimension $n \times n$, from section 4.4. Recall that $\mathbf{A} = \mathbf{A}^{\mathsf{T}}$ and $\mathbf{A}\mathbf{A} = \mathbf{A}$. Note that we can write the residuals as

$$\hat{\boldsymbol{\epsilon}} = \mathbf{y} - \hat{\boldsymbol{\mu}} = \mathbf{y} - \mathbf{A}\mathbf{y} = (\mathbf{I}_n - \mathbf{A})\mathbf{y}.$$

We therefore have

$$\begin{aligned}\mathbf{V}_{\hat{\boldsymbol{\epsilon}}} = \text{var}\{(\mathbf{I}_n - \mathbf{A})\mathbf{y}\} &= (\mathbf{I}_n - \mathbf{A})\text{var}(\mathbf{y})(\mathbf{I}_n - \mathbf{A})^\mathsf{T} \\ &= (\mathbf{I}_n - \mathbf{A})(\sigma^2\mathbf{I}_n)(\mathbf{I}_n - \mathbf{A})^\mathsf{T} \\ &= (\mathbf{I}_n - \mathbf{A})(\mathbf{I}_n - \mathbf{A})^\mathsf{T}\sigma^2 \\ &= (\mathbf{I}_n - \mathbf{A})(\mathbf{I}_n - \mathbf{A})\sigma^2 \\ &= (\mathbf{I}_n - \mathbf{A} - \mathbf{A} + \mathbf{A}\mathbf{A})\sigma^2 \\ &= (\mathbf{I}_n - \mathbf{A})\sigma^2.\end{aligned}$$

Further, because $\hat{\boldsymbol{\epsilon}}$ is a linear transformation of a multivariate normal random vector $(\mathbf{y})$, it must have a normal distribution

$$\hat{\boldsymbol{\epsilon}} \sim N(\mathbf{0}, (\mathbf{I}_n - \mathbf{A})\sigma^2).$$

We observe that although the errors may have equal variance and be uncorrelated, the residuals do not (as $\text{var}(\hat{\epsilon}_i) = (1 - A_{ii})\sigma^2$ and $\text{cov}(\hat{\epsilon}_i, \hat{\epsilon}_j) = -A_{ij}\sigma^2$, for $i \neq j$, where A_{ii} is the ith diagonal element of $\mathbf{A}$ and A_{ij} is its (i, j)th element). We can not do anything about the slight lack of independence, but the residuals can be standardized to have constant variance, by defining

$$\tilde{\epsilon}_i = \hat{\epsilon}_i / \sqrt{1 - A_{ii}}, \quad i = 1, \ldots, n.$$

Note that

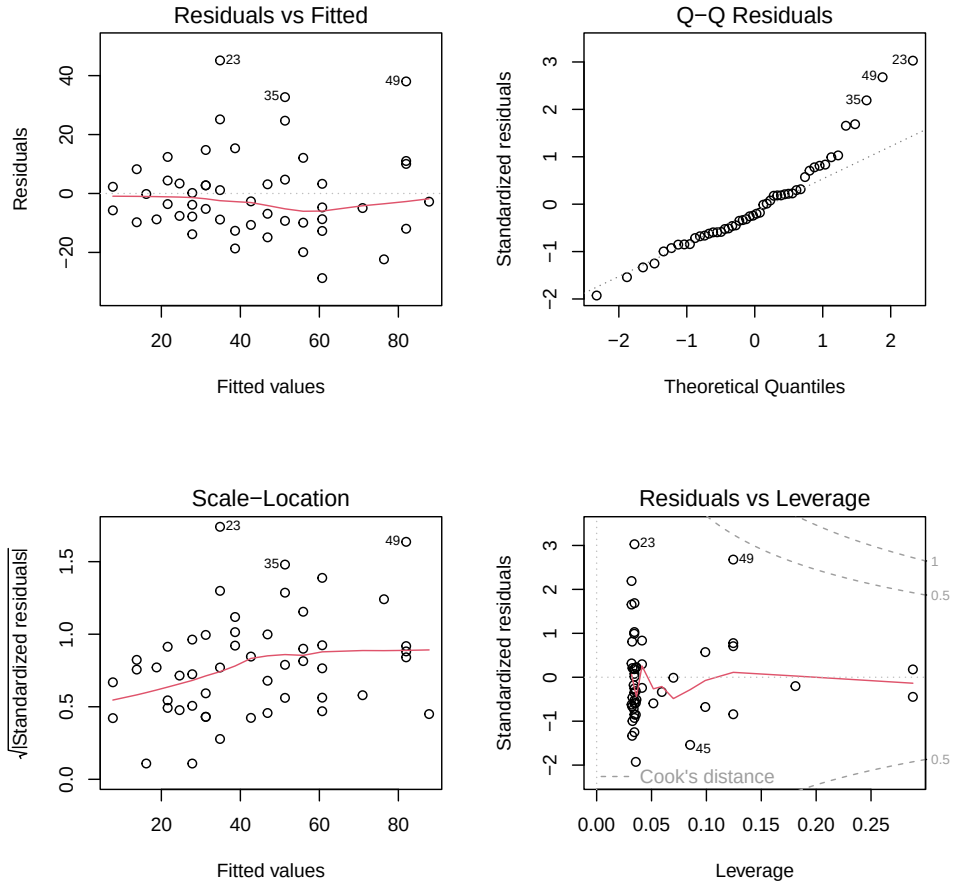
$$\text{var}(\tilde{\epsilon}_i) = \text{var}\left(\frac{\hat{\epsilon}_i}{\sqrt{1 - A_{ii}}}\right) = \frac{1}{1 - A_{ii}}\text{var}(\hat{\epsilon}_i) = \frac{(1 - A_{ii})\sigma^2}{1 - A_{ii}} = \sigma^2.$$

Further note that $\frac{\tilde{\epsilon}_i}{\sigma} \sim N(0, 1)$ if the model assumptions are met and the correlation between $\frac{\tilde{\epsilon}_i}{\sigma}$ and $\frac{\tilde{\epsilon}_j}{\sigma}$, $i \neq j$, tend to be small.

The way to check residuals is to examine a variety of diagnostic plots, always looking for evidence of any violation of the linear modelling assumptions. As Chambers et a.l (1983, p. 1) put it: "*There is no single statistical tool that is as powerful as a well-chosen graph.*" The `plot` function will provide a number of useful residual plots if passed a fitted linear model object. Let us revisit the cars dataset again and consider the running example model

$$\texttt{dist}_i = \beta_1 + \beta_2\texttt{speed}_i + \beta_3\texttt{speed}_i^2 + \epsilon_i, \quad \epsilon_i \text{ i.i.d. } N(0, \sigma^2).$$

```
data(cars)
cm <- lm(dist ~ speed + I(speed^2), data = cars)
par(mfrow = c(2, 2))
plot(cm)
```



- The upper left plot shows the model residuals, $\hat{\epsilon}_i$, against the model fitted values, $\hat{\mu}_i$, where $\hat{\boldsymbol{\mu}} = \mathbf{X}\hat{\boldsymbol{\beta}}$ and $\hat{\boldsymbol{\epsilon}} = \mathbf{y} - \hat{\boldsymbol{\mu}}$. The residuals should be evenly scattered above and below zero, independent of the x-axis positions (the distribution of fitted values is not of interest). The red line, which is a LOWESS (scatterplot smoother) fit regressing the residuals on the fitted values, should be constant around zero. A trend in the mean of the residuals is a sign of remaining systematic structure and usually results from an erroneous model structure (assumption 1.): e.g., assuming a linear relationship with a predictor, when a quadratic is required, or omitting an important predictor variable. A plot of the residuals against the predictors and potential predictors can help in further investigating/diagnosing the misspecification of the model mean structure. For plots of the residuals against the predictors not included in the model, any observed structure may indicate that this predictor should be included in the model.

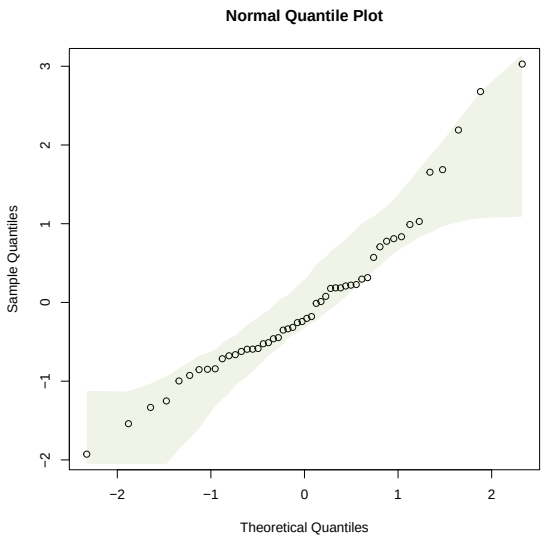
  This plot also allows investigating the assumption of constant variance. A trend in the variability of the residuals (e.g., a funnel shape) suggests that the variance of the response is related to its mean, violating the constant variance assumption (*heteroscedasticity*): transformation of the response or use of a generalised linear model may help, in such cases. The plot shown shows some indication of increasing variance with mean, which would somewhat violates the constant variance assumption, although the effect is not extreme here.

- The lower left plot is a scale-location plot. The raw residuals are standardized by dividing by their estimated standard deviation, $\hat{\sigma}\sqrt{1 - A_{ii}}$ ($\mathbf{A}$ is the influence matrix). The square root of the absolute value of each standardized residual is then plotted against the corresponding fitted value. It can be easier to judge the constant variance assumption from such a plot, and the square root transformation reduces the skew in the distribution, which would otherwise be likely to occur (note that if the standardised residuals follow a

normal distribution, the random variable given by their absolute value follows a 'half normal' distribution). We expect the red line to be almost constant about one. As the previous plot already hinted, there is some suggestion of variance increasing with mean in this plot, but the effect is not very large, so is unlikely to cause a big problem.

- The upper right panel is a normal Q-Q (quantile-quantile) plot. The standardized residuals are sorted and then plotted against the quantiles of a standard normal distribution. If the residuals are normally distributed then the resulting plot should look like a straight line relationship, perturbed by some random scatter. In this case the few large residuals show signs of being a bit too big to have come from the same normal distribution as the others, but again, the effect is not large relative to what can occur by chance even when the model assumptions are fine. Due to the difficulty in judging what deviation of a straight line can still be tolerated, there are alternative implementations which also plot a corresponding envelope, giving an idea of the natural variation. One approach is implemented in the function `qqenvelope` of the package `ecostats`.

```
library(ecostats)
qqenvelope(cm)
```



- The lower right panel plots the standardized residuals against the *leverage* of each datum. If a diagonal entry A_{ii} of the influence matrix is large, changing the corresponding response observation y_i will move the fitted line/surface appreciably towards the altered value. For this reason, A_{ii} is said to measure the leverage of the observation y_i. Observations that are outliers in the predictor space are known as high leverage points, to distinguish them from observations that are outliers in the response variable (those with large, in absolute value, standardised residuals). The trace of the matrix $\mathbf{A}$ is p. Why? Remember that for two matrices $\mathbf{U_1}$, of dimension, $n \times p$, and $\mathbf{U_2}$, of dimension, $p \times n$, $\text{tr}(\mathbf{U_1 U_2}) = \text{tr}(\mathbf{U_2 U_2})$. Therefore, $\text{tr}(\mathbf{A}) = \text{tr}(\mathbf{Q Q}^\mathsf{T}) = \text{tr}(\mathbf{Q}^\mathsf{T}\mathbf{Q}) = \text{tr}(\mathbf{I}_p) = p$. The dimension of the model space is also p, and so large leverage is taken to be greater than two or three times the average p/n (recall that $\mathbf{A}$ is of dimension $n \times n$ and so there are n diagonal elements). The combination of a large residual and high leverage implies that the corresponding datum has a substantial influence on the overall fit. Large leverage with a small residual generally implies that, although the datum would exert undue influence on the fit, if out of line with the other data, it is actually consistent with the other data, and does not have too great an influence. Similarly a large residual may not have much influence on the fit, if the corresponding datum has low leverage. This is illustrated in figure 5.

  A good summary of how much influence each datum actually has is provided by its *Cook's distance*. Cook's distance is a measure of how much influence each observation has on the fitted model. If $\hat{\mu}_i^{[k]}$ is the i^{th} fitted

value, when the k^{th} datum is omitted from the fit, then Cook's distance is

$$d_k = \frac{1}{(p+1)\hat{\sigma}^2} \sum_{i=1}^{n} (\hat{\mu}_i^{[k]} - \hat{\mu}_i)^2, \tag{13}$$

where p is the number of parameters and n the number of data points. Cook's distance can be shown to be a function of leverage and standardized residual, so contours of Cook's distance are shown on the plot. A very large value of d_k indicates a point that has a substantial influence on the model results. Cook's distances above 0.5 are considered borderline problematic, whereas values over 1 are usually considered highly influential. If the Cook's distance values indicate that model estimates may be very sensitive to just one or two data, then it usually prudent to repeat any analysis without the offending points, in order to check the robustness of the modelling conclusions. In this case none of the points look wildly out of line.

The leverage and Cook's distances can also be easily obtained from the model fit.

```
leverage <-  hatvalues(cm)
head(leverage)
```

```
##          1          2          3          4          5          6
## 0.28812937 0.28812937 0.09900328 0.09900328 0.06991940 0.05169124
```

```
#any observations with high or very high leverage?
as.numeric(which(leverage > 2*(3/nrow(cars))))
```

```
## [1]  1  2 46 47 48 49 50
```

```
as.numeric(which(leverage > 3*(3/nrow(cars))))
```

```
## [1]  1  2 50
```

```
# Cook's distances and any above 0.5
cooks_dist <- cooks.distance(cm)
as.numeric(which(cooks_dist > 0.5))
```

```
## numeric(0)
```

```
# We can also plot observation number against Cook's distance
plot(cm, which = 4)
```

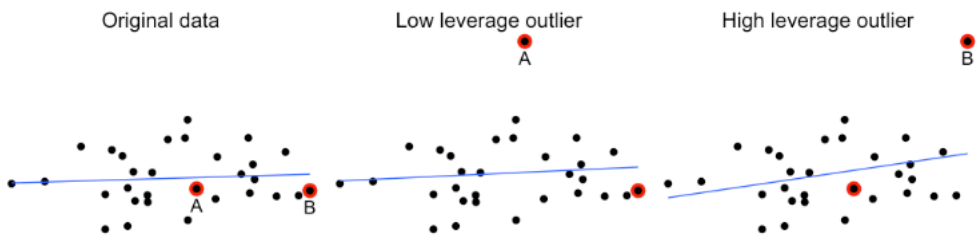
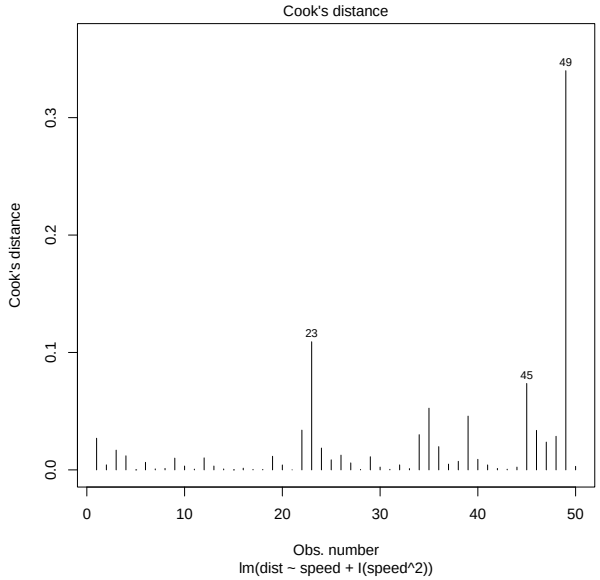


Figure 5: High leverage point B has a relatively large effect on the regression line. Figure taken from Speegle and Clair (2022, p. 394)

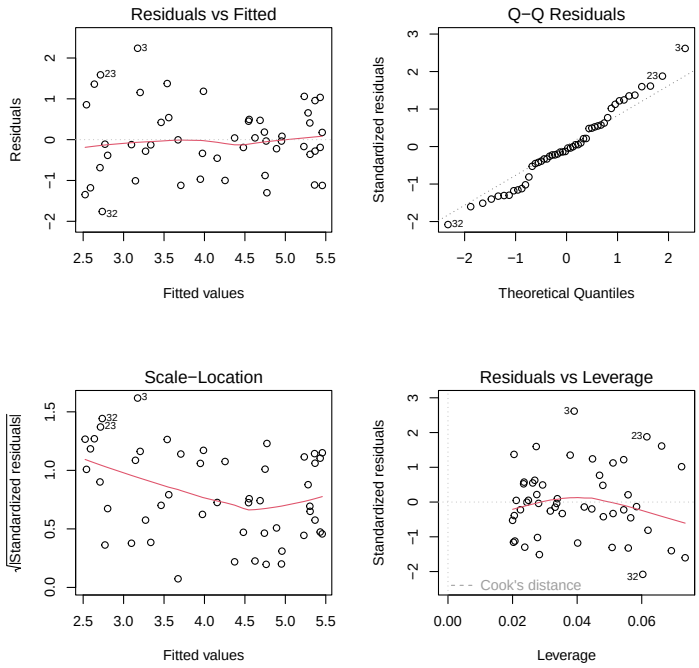


- Finally, we address the issue of correlated errors, which is difficult to check for in general because there are just too many possible patterns of correlation that may occur. But some types of data have a structure which suggest where to look for problems. For instance, data collected over time may have some correlation in successive errors, whereas spatial data may have correlation in the errors of nearby measurements. The presence of strong dependence means that there is less information in the data than the sample size may suggest. For the case of temporal data, one possible approach to detect serial correlation is to plot the residuals against time and check if there are any patterns (if there are, this is a sign of correlation). An alternative approach is to plot successive pairs of residuals. Methods developed for time series may be used if there is temporal correlation but this is out of the scope of this course.

It is important to realize that residual plots are quite variable, even when all the model assumptions are met. To get a feel for this it can be helpful to simulate some residuals for cases where the assumptions are fine...

```
set.seed(6) #ensures exact repeatability
n <- 50 #sample size
x <- runif(n) #simulate a uniform(0,1) predictor
eps <- rnorm(n) #simulate a normal error
y <- 2 + 4*x + eps #response variable
fit <- lm(y ~ x) #fit a linear model to data
```

```r
par(mfrow = c(2, 2))
plot(fit)
```



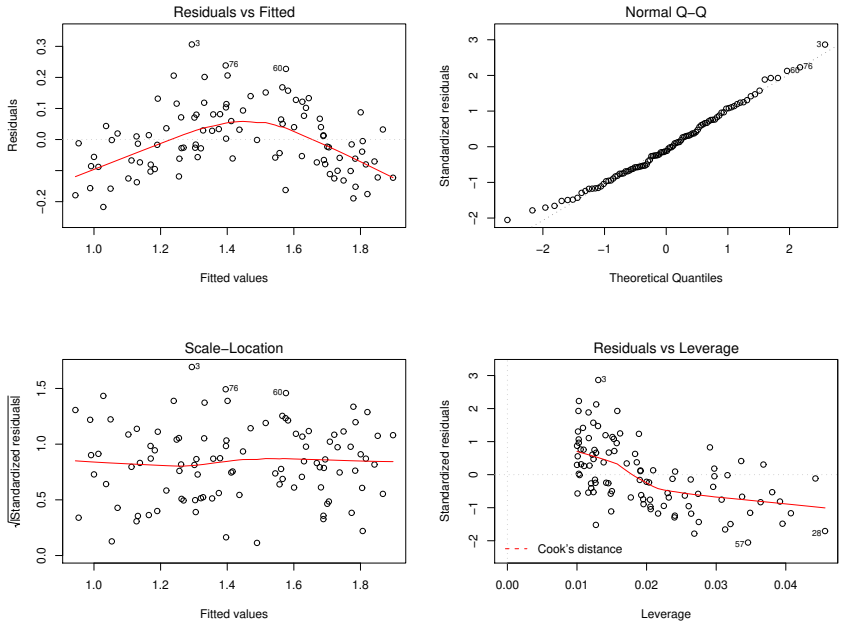
Repeating this simulation exercise a few times (without the `set.seed` line, of course), will give you a feel for the degree of variability to expect when the model assumptions are ok. Note that there is a sample size dependence in this variability! Further note that the red line is a flexible fit and, as such, it is extremely sensitive to areas where data are scarce.

5.1.4 Residual plots: common problems

When looking at residual plots, the most important things to check for are

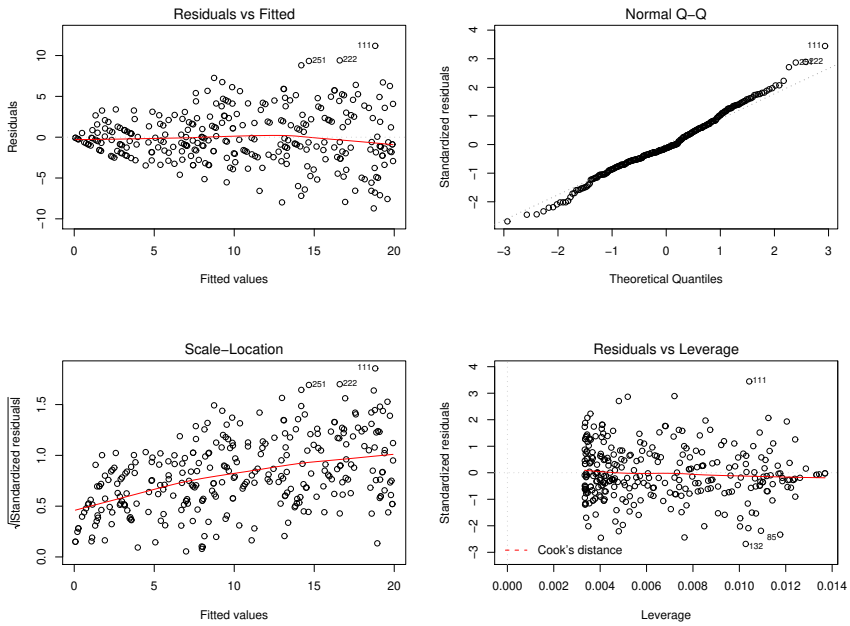
1. Anything that suggests that the residuals might not all have zero expected value.
2. Anything that suggests that the residuals might not all have the same variance.

Here is a problematic set of plots.

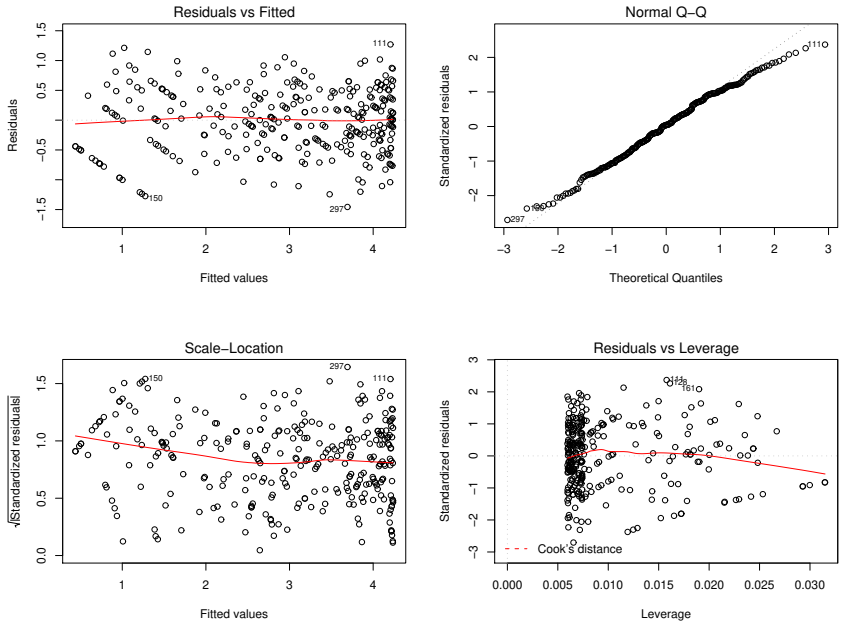


Notice the clear trend in the mean of the residuals plotted against fitted values. The mean of the residuals is below zero for low and high fitted values and above zero for intermediate fitted values. This sort of pattern means that something is missing from the structural part model of the model. Perhaps a dependence on the square of some predictor variable?

Here is a set of plots with a different problem. . .

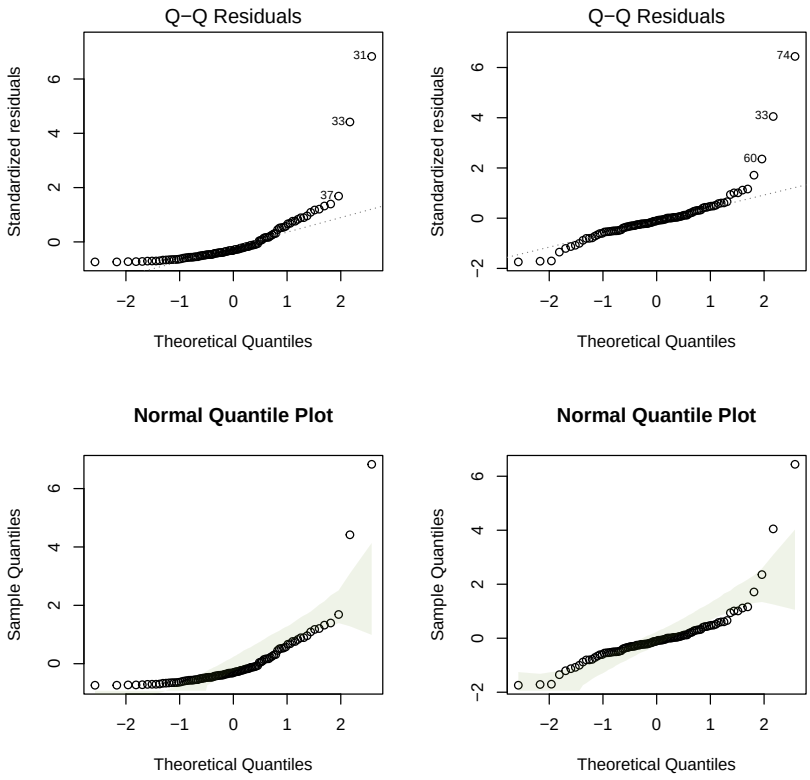


In this case the mean seems to be more or less zero for all the residuals, as it should be. However the variance of the residuals shows a clear increase with the mean. This pattern of increasing variance with increasing mean is quite a common pattern in practice, and obviously violates the constant variance assumption. In this case it can sometimes help to switch to modelling some transformation of the original response variable (which, of course, in this case did not take negative values). For example the following plots are from a model of the *square root* of the response variable from the previous plot.



Finally, in order to get an idea of the variation to be expected in Q–Q plots we have simulated a dataset from a lognormal distribution, which is an example of a skewed distribution (left plot), and from a t distribution, which is a heavy tailed distribution (right plot). Envelopes plots are also shown. You can change the seed and observe a few more plots.

```
set.seed(6)
n <- 100
x <- runif(n)
eps <- rlnorm(n)
y <- 2 + 4*x + eps
fit_logn <- lm(y ~ x)

set.seed(6)
n <- 100
x <- runif(n)
eps <- rt(n, df = 3)
y <- 2 + 4*x + eps
fit_t <- lm(y ~ x)

par(mfrow = c(2, 2))
plot(fit_logn, which = 2)
plot(fit_t, which = 2)

qqenvelope(fit_logn)
qqenvelope(fit_t)
```



5.2 Tyre wear data and backward selection

The `MASS` library, which is an add on for R, contains a data set, `Rubber`, on the relationship between the rate of wear of tyres (`loss` in grammes per hour), and the hardness and tensile strength of the tyre rubber (`hard` and `tens` respectively, measured in 'Shore units' and in kgm^{-2}, respectively).

```
library(MASS)
head(Rubber)
```

```
##   loss hard tens
## 1  372   45  162
## 2  206   55  233
## 3  175   61  232
## 4  154   66  231
## 5  136   71  231
## 6  112   71  237
```

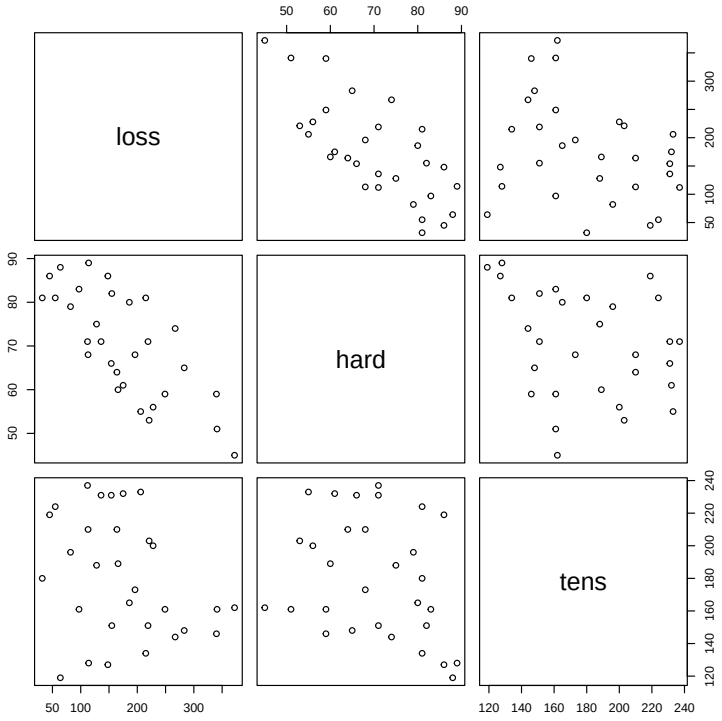
```
str(Rubber)
```

```
## 'data.frame': 30 obs. of  3 variables:
##  $ loss: int  372 206 175 154 136 112 55 45 221 166 ...
##  $ hard: int  45 55 61 66 71 71 81 86 53 60 ...
##  $ tens: int  162 233 232 231 231 237 224 219 203 189 ...
```

```
summary(Rubber)
```

```
##       loss            hard            tens
##  Min.   : 32.0   Min.   :45.00   Min.   :119.0
```

```
##  1st Qu.:113.2   1st Qu.:60.25   1st Qu.:151.0
##  Median :165.0   Median :71.00   Median :176.5
##  Mean   :175.4   Mean   :70.27   Mean   :180.5
##  3rd Qu.:220.5   3rd Qu.:81.00   3rd Qu.:210.0
##  Max.   :372.0   Max.   :89.00   Max.   :237.0

#scatter plot matrix of loss, hard, and tens
plot(Rubber)
```



The data were gathered in order to develop a model with which to predict wear rate as a function of hardness and tensile strength. All that is really known at the outset is that the loss rate should be some smooth function of the two predictors, so we could try, for instance, a polynomial model in which all combinations of powers of `tens` and `hard` were present, up to third order. That is, we could try the model:

$$y_i = \beta_0 + \beta_1 h_i + \beta_2 t_i + \beta_3 t_i h_i + \beta_4 t_i^2 + \beta_5 h_i^2 + \beta_6 h_i^2 t_i + \beta_7 h_i t_i^2 + \beta_8 t_i^3 + \beta_9 h_i^3 + \epsilon_i$$

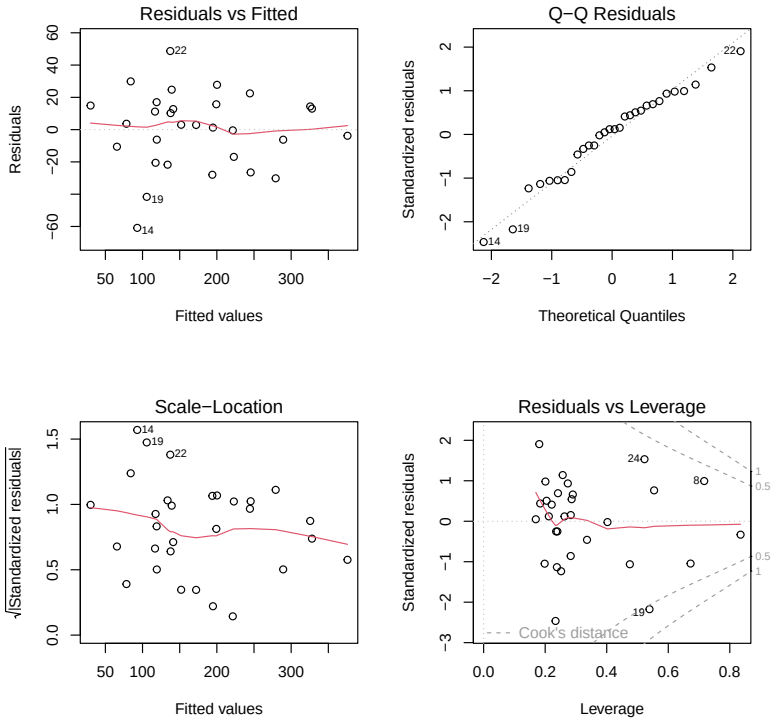
where y is `loss`, h is `hard`, t is `tens` and the ϵ_i are i.i.d. $N(0, \sigma^2)$ random variables. In the interests of precision it is worth trying to see whether all the variables in this model are needed, or whether something simpler would suffice. Note that the sample size n is 30 and the model has ten parameters. To that end we can successively remove the variable for which the hypothesis of zero value yields the highest p-value, above some threshold (e.g. 0.05), until all variables in the model have p-values below the threshold.

Here is how this *backward selection* can be achieved in R. We start by fitting the hypothesised model and by inspecting the residuals plots.

```
m1 <- lm(loss ~ hard + tens + I(hard*tens) + I(hard^2) + I(tens^2) +
                 I(hard^2*tens) + I(tens^2*hard) + I(tens^3) + I(hard^3),
         data = Rubber)

summary(m1)

##
```

```
## Call:
## lm(formula = loss ~ hard + tens + I(hard * tens) + I(hard^2) +
##     I(tens^2) + I(hard^2 * tens) + I(tens^2 * hard) + I(tens^3) +
##     I(hard^3), data = Rubber)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -60.880 -15.307   2.958  14.784  48.695
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -6.040e+02  4.543e+03  -0.133    0.896
## hard             -1.240e+01  7.213e+01  -0.172    0.865
## tens              3.123e+01  5.295e+01   0.590    0.562
## I(hard * tens)   -2.924e-01  5.085e-01  -0.575    0.572
## I(hard^2)         4.968e-01  7.613e-01   0.653    0.521
## I(tens^2)        -1.572e-01  2.081e-01  -0.755    0.459
## I(hard^2 * tens)  2.469e-03  1.675e-03   1.474    0.156
## I(tens^2 * hard) -7.527e-05  9.350e-04  -0.081    0.937
## I(tens^3)         3.384e-04  2.788e-04   1.214    0.239
## I(hard^3)        -4.736e-03  3.897e-03  -1.215    0.238
##
## Residual standard error: 28.22 on 20 degrees of freedom
## Multiple R-squared:  0.9292,Adjusted R-squared:  0.8973
## F-statistic: 29.16 on 9 and 20 DF,  p-value: 1.373e-09
```

```
par(mfrow = c(2, 2))
plot(m1)
```



For only 30 data points, these plots look ok. It might be worth repeating the analysis with and without observation 19, which has a high Cook's distance and may therefore be having undue influence on the results. We shall proceed

and remove the variable with the highest p-value, which is ht^2.

```
m2 <- update(m1, . ~ . - I(tens^2*hard))
summary(m2)

##
## Call:
## lm(formula = loss ~ hard + tens + I(hard * tens) + I(hard^2) +
##     I(tens^2) + I(hard^2 * tens) + I(tens^3) + I(hard^3), data = Rubber)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -60.298 -14.934   3.096  15.266  48.709
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -9.349e+02  1.890e+03  -0.495   0.6261
## hard             -8.602e+00  5.324e+01  -0.162   0.8732
## tens              3.520e+01  1.910e+01   1.843   0.0795 .
## I(hard * tens)   -3.299e-01  1.993e-01  -1.655   0.1127
## I(hard^2)         4.927e-01  7.414e-01   0.665   0.5136
## I(tens^2)        -1.715e-01  1.060e-01  -1.618   0.1207
## I(hard^2 * tens)  2.536e-03  1.419e-03   1.787   0.0884 .
## I(tens^3)         3.545e-04  1.906e-04   1.860   0.0770 .
## I(hard^3)        -4.780e-03  3.766e-03  -1.269   0.2182
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.55 on 21 degrees of freedom
## Multiple R-squared:  0.9292,Adjusted R-squared:  0.9022
## F-statistic: 34.44 on 8 and 21 DF,  p-value: 2.237e-10
```

We now remove the variable hard, h.

```
m3 <- update(m2, . ~ . - hard)
summary(m3)

##
## Call:
## lm(formula = loss ~ tens + I(hard * tens) + I(hard^2) + I(tens^2) +
##     I(hard^2 * tens) + I(tens^3) + I(hard^3), data = Rubber)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -60.074 -15.725   2.946  14.057  48.952
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -1.185e+03  1.061e+03  -1.116   0.2764
## tens              3.630e+01  1.742e+01   2.084   0.0490 *
## I(hard * tens)   -3.436e-01  1.763e-01  -1.949   0.0642 .
## I(hard^2)         3.803e-01  2.516e-01   1.512   0.1448
## I(tens^2)        -1.752e-01  1.012e-01  -1.731   0.0975 .
## I(hard^2 * tens)  2.638e-03  1.245e-03   2.118   0.0457 *
## I(tens^3)         3.612e-04  1.818e-04   1.987   0.0595 .
## I(hard^3)        -4.308e-03  2.323e-03  -1.854   0.0772 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

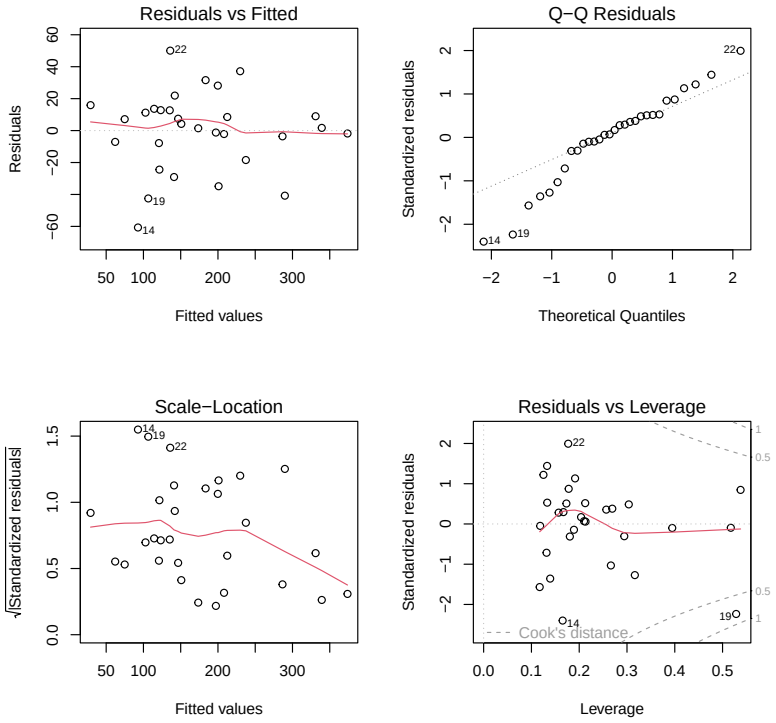
```
##
## Residual standard error: 26.93 on 22 degrees of freedom
## Multiple R-squared:  0.9291,Adjusted R-squared:  0.9065
## F-statistic: 41.18 on 7 and 22 DF,  p-value: 3.356e-11
```

We now remove the variable h^2.

```
m4 <- update(m3, . ~ . - I(hard^2))
summary(m4)

##
## Call:
## lm(formula = loss ~ tens + I(hard * tens) + I(tens^2) + I(hard^2 *
##     tens) + I(tens^3) + I(hard^3), data = Rubber)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -60.690  -7.647   2.943  12.765  50.057
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -1.252e+03  1.090e+03  -1.148 0.262582
## tens              3.810e+01  1.786e+01   2.133 0.043794 *
## I(hard * tens)   -8.193e-02  3.443e-02  -2.379 0.026016 *
## I(tens^2)        -2.319e-01  9.656e-02  -2.402 0.024789 *
## I(hard^2 * tens)  8.105e-04  3.090e-04   2.623 0.015201 *
## I(tens^3)         4.574e-04  1.750e-04   2.614 0.015507 *
## I(hard^3)        -8.090e-04  2.072e-04  -3.905 0.000713 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.67 on 23 degrees of freedom
## Multiple R-squared:  0.9217,Adjusted R-squared:  0.9013
## F-statistic: 45.14 on 6 and 23 DF,  p-value: 1.375e-11
```

All variables in this last model are now significant at the 5% level. In the above the `update` function was used to *update* the fitted model: by specifying only the variables to be dropped from the previous fit there is less typing to do (compared to calling `lm` each time), and hence less chance of error. Here are the final residual plots.

```
par(mfrow = c(2, 2))
plot(m4)
```



As before, for only 30 observations this look reasonable. As before, observation 19 has a high Cook's distance and might be worth investigating its impact on both model fit and model selection.

5.2.1 `drop1`

`drop1` is a useful R function which will work through the terms in your model, dropping each term and comparing the fit of the model without that term, to the full model fit. For a linear model the comparison can be done by F ratio testing or AIC (see below). For example, for the full tyre wear model we get the following.

```
drop1(m1, test = "F")

## Single term deletions
##
## Model:
## loss ~ hard + tens + I(hard * tens) + I(hard^2) + I(tens^2) +
##     I(hard^2 * tens) + I(tens^2 * hard) + I(tens^3) + I(hard^3)
##                  Df Sum of Sq   RSS    AIC F value Pr(>F)
## <none>                        15932 208.25
## hard              1     23.55 15955 206.29  0.0296 0.8652
## tens              1    277.24 16209 206.76  0.3480 0.5618
## I(hard * tens)    1    263.52 16195 206.74  0.3308 0.5716
## I(hard^2)         1    339.26 16271 206.88  0.4259 0.5214
## I(tens^2)         1    454.44 16386 207.09  0.5705 0.4589
## I(hard^2 * tens)  1   1730.76 17662 209.34  2.1728 0.1560
## I(tens^2 * hard)  1      5.16 15937 206.25  0.0065 0.9366
## I(tens^3)         1   1173.63 17105 208.38  1.4733 0.2390
## I(hard^3)         1   1176.52 17108 208.38  1.4770 0.2384
```

The p-value for each row is that obtained by testing the null hypothesis that the model *without* that row's term is correct, against the alternative that the full model is required. So a high p-value implies that the data are quite consistent with the model in which the term concerned has been dropped. Notice that because each model

comparison here involves just one parameter, the p-values obtained by F ratio testing are the same as those obtained from the t-tests used in `summary`. So why bother with `drop1`? The answer is that it is much more convenient than `summary` when there are factor variables in the model. In such a case `summary` will report p-values for each parameter associated with the factor, which are not suitable for deciding on whether the whole factor variable should be included in the model or not. By contrast, `drop1` will compare the models with and without the factor variable (thereby dealing with all its parameters at once) which is what is needed for model selection.

5.2.2 AIC

The default model comparison method for `drop1` is AIC: Akaike's Information Criterion . When we compare models by hypothesis testing we are basically asking 'what is the simplest model that is defensible for these data', since hypothesis testing always sticks with the simplest model unless the data provide strong evidence that it is less adequate than the alternative. AIC is a different approach, it is an attempt to estimate a particular measure of the 'distance' between the fitted model, and the underlying 'true model' that generated the data. If we do model selection based on the AIC then we are trying to find the the model that will be as 'close' as possible to the truth, in the sense of having the smallest error when the model is used for prediction. Models with lower AIC are considered to be better than models with higher AIC. Formally if l is the maximized log likelihood of the model and p the number of parameters one has then

$$\text{AIC} = -2l + 2p.$$

For a linear regression model with iid $N(0, \sigma^2)$ errors, fitted to n data points, it is easy to show that

$$\text{AIC} = n \log\left(\frac{\text{RSS}}{n}\right) + C + 2p,$$

where $C = n + n\log(2\pi)$ is a constant that depends only on the sample size and not on the model and, as such, for variable selection purposes can be ignored. Adding more predictors to a model cannot increase the RSS but one is discouraged of adding too many because of the term $2p$, called penalty term. Hence the AIC naturally provides a balance between fit and simplicity in model selection. The `AIC` function will compute the AIC of a fitted model object in R.

```
AIC(m1)

## [1] 295.382
```

And here is what `drop1` does by default.

```
drop1(m1)

## Single term deletions
##
## Model:
## loss ~ hard + tens + I(hard * tens) + I(hard^2) + I(tens^2) +
##     I(hard^2 * tens) + I(tens^2 * hard) + I(tens^3) + I(hard^3)
##                   Df Sum of Sq   RSS    AIC
## <none>                         15932 208.25
## hard               1     23.55 15955 206.29
## tens               1    277.24 16209 206.76
## I(hard * tens)     1    263.52 16195 206.74
## I(hard^2)          1    339.26 16271 206.88
## I(tens^2)          1    454.44 16386 207.09
## I(hard^2 * tens)   1   1730.76 17662 209.34
## I(tens^2 * hard)   1      5.16 15937 206.25
## I(tens^3)          1   1173.63 17105 208.38
## I(hard^3)          1   1176.52 17108 208.38
```

The row labelled `<none>` gives the AIC of the full model (model `m1` in this case). Irritatingly, this differs from that given by `AIC` because of differences in which model independent constants have been included in the

log likelihood of the model: of course differences in AIC are unaffected by such details, as already mentioned. The remaining rows report the AIC of the model without the term concerned (but with all other model terms). The lowest AIC is for the model with `I(tens^2 * hard)` dropped, so this would be the first candidate for dropping.

5.2.3 `step`

Backwards selection by iteratively calling `drop1`, and deleting the term it implies is rather automatic, and could clearly be automated completely. Function `step` does this, based on AIC model selection. `step` actually does slightly more than backwards selection, since at each stage (after the second), it tries to see whether the model could be improved by re-including any of the currently dropped terms, before it considers further deletions. Here is the outcome of applying step to the tyre model.

```
step(m1)
```

```
## Start:  AIC=208.25
## loss ~ hard + tens + I(hard * tens) + I(hard^2) + I(tens^2) +
##     I(hard^2 * tens) + I(tens^2 * hard) + I(tens^3) + I(hard^3)
##
##                     Df Sum of Sq   RSS    AIC
## - I(tens^2 * hard)  1      5.16 15937 206.25
## - hard              1     23.55 15955 206.29
## - I(hard * tens)    1    263.52 16195 206.74
## - tens              1    277.24 16209 206.76
## - I(hard^2)         1    339.26 16271 206.88
## - I(tens^2)         1    454.44 16386 207.09
## <none>                          15932 208.25
## - I(tens^3)         1   1173.63 17105 208.38
## - I(hard^3)         1   1176.52 17108 208.38
## - I(hard^2 * tens)  1   1730.76 17662 209.34
##
## Step:  AIC=206.26
## loss ~ hard + tens + I(hard * tens) + I(hard^2) + I(tens^2) +
##     I(hard^2 * tens) + I(tens^3) + I(hard^3)
##
##                     Df Sum of Sq   RSS    AIC
## - hard              1     19.81 15956 204.29
## - I(hard^2)         1    335.12 16272 204.88
## <none>                          15937 206.25
## - I(hard^3)         1   1222.59 17159 206.47
## - I(tens^2)         1   1985.73 17922 207.78
## - I(hard * tens)    1   2079.83 18016 207.94
## - I(hard^2 * tens)  1   2423.23 18360 208.50
## - tens              1   2577.65 18514 208.75
## - I(tens^3)         1   2624.71 18561 208.83
##
## Step:  AIC=204.29
## loss ~ tens + I(hard * tens) + I(hard^2) + I(tens^2) + I(hard^2 *
##     tens) + I(tens^3) + I(hard^3)
##
##                     Df Sum of Sq   RSS    AIC
## <none>                          15956 204.29
## - I(hard^2)         1    1657.6 17614 205.26
## - I(tens^2)         1    2173.1 18130 206.12
## - I(hard^3)         1    2493.9 18450 206.65
## - I(hard * tens)    1    2754.8 18711 207.07
## - I(tens^3)         1    2863.5 18820 207.24
## - tens              1    3150.1 19106 207.70
```

```
## - I(hard^2 * tens)  1    3253.1 19210 207.86
##
## Call:
## lm(formula = loss ~ tens + I(hard * tens) + I(hard^2) + I(tens^2) +
##     I(hard^2 * tens) + I(tens^3) + I(hard^3), data = Rubber)
##
## Coefficients:
##      (Intercept)              tens    I(hard * tens)         I(hard^2)
##       -1.185e+03         3.630e+01        -3.436e-01         3.803e-01
##        I(tens^2)  I(hard^2 * tens)          I(tens^3)         I(hard^3)
##       -1.752e-01         2.637e-03          3.612e-04        -4.308e-03
```

Notice that this model is different to the one we arrived at by hypothesis testing, and has more terms. AIC generally tends to select more complex models than hypothesis testing, because of its different objective: the 'best' model, in a prediction error sense, is likely to be more complicated than 'the simplest model that the data will allow'.

5.3 Forward and forward-backward selection

In the tyre wear example, it was fairly easy to decide to start with a cubic model and use backward selection. Given the aim of finding the optimum rubber we needed at least quadratic model, and quartic would have been excessively complex for the number of data. Backward selection is also nice theoretically. We start with a model that is complicated enough that we are fairly sure that it is not wrong, it is just that it is overcomplicated. By starting with a more or less correct model, we ensure that the distributional assumptions underpinning the hypothesis tests (or AIC computation) are valid, so that the whole procedure is well founded.

However, it is often the case that there is no obvious candidate for the 'most complex reasonable model' or that such a model would be far too complicated to use in practice. In this case we might choose to start with a relatively simple model (e.g. some initial subset of variables enter as simple linear effects or even a model only including an intercept) hoping that it is not too far wrong. We then repeatedly use hypothesis testing or AIC comparison to consider adding in extra variables, or higher order terms, one at a time. If an added term improves the fit significantly we keep it in the model. This method is 'forward selection'. Clearly in using this method we are starting from a model that we are accepting is wrong, and if the procedure proceeds for more than one step, then it must rely on comparing models when both of them are wrong (we have no rigorous theory for this case). Nonetheless we can often end up with an adequate model this way, in which case the theoretical problems with the method used to find it are often unimportant.

Forward and backward selection can be combined in various ways. For example, run forward selection until no further variables are added, then run backward selection until no further variables are dropped, then run forward selection again, and so on until successive forward and backward steps both produce no change in the model.

The `step` function can perform backward, forward, or forward-backward selection. For example, here is the tyre example again, but this time starting from a model with simple linear terms in `hard` and `tens`.

```
m0 <- lm(loss ~ hard + tens, data = Rubber)
step(m0, scope = loss ~ hard + tens + I(hard*tens) + I(hard^2) + I(tens^2)+
       I(hard^2*tens) + I(tens^2*hard) + I(tens^3) + I(hard^3),
     direction = "both")

## Start:  AIC=218.66
## loss ~ hard + tens
##
##                     Df Sum of Sq    RSS    AIC
## + I(tens^2 * hard)   1      5971  29978 215.21
## + I(hard * tens)     1      2919  33031 218.12
## <none>                           35950 218.66
## + I(hard^3)          1      2047  33903 218.90
## + I(hard^2)          1      1723  34227 219.19
## + I(tens^3)          1      1536  34414 219.35
## + I(hard^2 * tens)   1      1458  34492 219.42
```

```
## + I(tens^2)         1      1042  34907 219.78
## - tens              1     66607 102556 248.11
## - hard              1    169027 204977 268.88
##
## Step:  AIC=215.21
## loss ~ hard + tens + I(tens^2 * hard)
##
##                    Df Sum of Sq   RSS    AIC
## + I(hard^3)         1      2099 27879 215.03
## <none>                          29978 215.21
## + I(hard^2)         1      1914 28065 215.23
## + I(hard^2 * tens)  1      1556 28423 215.61
## + I(tens^3)         1      1382 28596 215.79
## + I(tens^2)         1      1005 28973 216.19
## + I(hard * tens)    1       329 29649 216.88
## - I(tens^2 * hard)  1      5971 35950 218.66
## - tens              1     12181 42159 223.44
## - hard              1     39189 69168 238.29
##
## Step:  AIC=215.03
## loss ~ hard + tens + I(tens^2 * hard) + I(hard^3)
##
##                    Df Sum of Sq   RSS    AIC
## + I(tens^3)         1    2279.0 25600 214.47
## + I(tens^2)         1    1873.5 26005 214.95
## <none>                          27879 215.03
## - I(hard^3)         1    2099.5 29978 215.21
## + I(hard * tens)    1    1078.2 26801 215.85
## - hard              1    2753.4 30632 215.86
## + I(hard^2)         1     989.0 26890 215.95
## + I(hard^2 * tens)  1     596.5 27282 216.38
## - I(tens^2 * hard)  1    6024.2 33903 218.90
## - tens              1   12776.0 40655 224.35
##
## Step:  AIC=214.47
## loss ~ hard + tens + I(tens^2 * hard) + I(hard^3) + I(tens^3)
##
##                    Df Sum of Sq   RSS    AIC
## + I(hard^2 * tens)  1    8638.1 16962 204.13
## + I(hard * tens)    1    7164.8 18435 206.62
## + I(tens^2)         1    6523.7 19076 207.65
## + I(hard^2)         1    2988.1 22612 212.75
## <none>                          25600 214.47
## - hard              1    2013.1 27613 214.75
## - I(tens^3)         1    2279.0 27879 215.03
## - I(hard^3)         1    2996.3 28596 215.79
## - I(tens^2 * hard)  1    5835.1 31435 218.63
## - tens              1   13803.3 39403 225.41
##
## Step:  AIC=204.13
## loss ~ hard + tens + I(tens^2 * hard) + I(hard^3) + I(tens^3) +
##     I(hard^2 * tens)
##
##                    Df Sum of Sq   RSS    AIC
## <none>                          16962 204.13
## + I(hard^2)         1     575.1 16387 205.09
## + I(tens^2)         1     361.7 16600 205.48
```

```
## + I(hard * tens)    1       2.7 16959 206.12
## - hard              1    4172.0 21134 208.72
## - I(tens^2 * hard)  1    6317.7 23279 211.62
## - I(hard^2 * tens)  1    8638.1 25600 214.47
## - I(tens^3)         1   10320.6 27282 216.38
## - I(hard^3)         1   11380.9 28343 217.53
## - tens              1   20802.6 37764 226.14
##
## Call:
## lm(formula = loss ~ hard + tens + I(tens^2 * hard) + I(hard^3) +
##     I(tens^3) + I(hard^2 * tens), data = Rubber)
##
## Coefficients:
##      (Intercept)              hard              tens  I(tens^2 * hard)
##        1.780e+03        -7.546e+00        -8.282e+00        -7.584e-04
##        I(hard^3)         I(tens^3)  I(hard^2 * tens)
##       -2.036e-03         1.433e-04         2.248e-03
```

Slightly different to the previous model.

5.4 Remarks on model selection

Finally let's review the reasons for model selection.

1. We do model selection because we are often uncertain about the exact form that a model should take, even though it is often possible to write down a model that we expect to be 'complicated enough', so that that for some parameter values it should be a reasonable approximation to the truth.
2. Selection is important for interpretational reasons: simpler models are easy to interpret than complex ones.
3. Model selection also tends to improve the precision of estimates and the accuracy of model predictions.

Whether model selection is performed using AIC or hypothesis testing depends on the purposes of the analysis. If we want to develop a model for prediction purposes then it makes sense to use AIC, but if our interest lies in trying to understand relationships between the predictors and the response, it may be preferable to use hypothesis testing based methods to try and avoid including model terms unless there is good evidence that they are needed.

Finally note that there is a difficult problem associated with model selection:

- It is common practice to use model selection methods to choose one model from a large set of potential models, but then to treat the selected model exactly as if it were the only model we ever considered, when it comes to calculating confidence intervals etc. In doing this we neglect the uncertainty associated with model selection, and will therefore tend to overstate how precisely we know the coefficients of the selected model (and how precise its predictions are). This issue is an active area of current statistical research.

5.5 r^2: How close is the fit?

It is useful to have a general measure of how closely a model fits the response data, and the r^2 statistic provides this. r^2 measures the proportion (or percentage) of the variance in the original data that is 'explained' by the fitted model. The proportion of variance left unexplained, is the observed variance of the residuals, divided by the observed variance of the response data. The proportion of variance explained is hence one minus the same ratio, so

$$r^2 = 1 - \frac{\sum_i \hat{\epsilon}_i^2/n}{\sum_i (y_i - \bar{y})^2/n}.$$

Note that the denominator can be thought as the observed variance of the residuals of a regression model that contains only an intercept term, $y_i = \beta + \epsilon_i$ (as we have seen in Workshop 1, $\hat{\beta} = \bar{y}$ and $\hat{\epsilon}_i = y_i - \bar{y}$). As we already observed when introducing the AIC, the residual sum of squares, when a predictor is added, can not

increase. Since the denominator depends only on $\{y_i\}_{i=1}^n$ and is identical to every model fitted to a particular dataset, we conclude that when predictors are added to a model, r^2 is monotone increasing. Also, for exactly this reason, we have that $\sum_i \hat{\epsilon}_i^2 \leq \sum_i (y_i - \bar{y})^2$ thus implying that $(\sum_i \hat{\epsilon}_i^2/n)/(\sum_i (y_i - \bar{y})^2/n) \leq 1$ and therefore, $0 \leq r^2 \leq 1$. Further notice that this conventional definition (from which the n's can be cancelled) uses biased variance estimators. As a result r^2 tends to overestimate how well a model is doing. The *adjusted* r^2 avoids this, to some extent, by using unbiased estimators:

$$r^2_{\text{adj}} = 1 - \frac{\sum_i \hat{\epsilon}_i^2/(n-p)}{\sum_i (y_i - \bar{y})^2/(n-1)},$$

where p is the number of model parameters. r^2_{adj} need not monotonically increase as predictors are added to a model. Note that r^2_{adj} can be negative, which occurs when an estimated model predicts worse than an intercept only model.

A high r^2 is always better than a low r^2 (provided that the model assumptions are met), but a low r^2 should not necessarily be taken as indicating that a model is poor: some response data simply contain a good deal of random variability about which nothing can be done. Note that r^2/r^2_{adj} does not measure the correctness of a linear model but its usefulness, assuming the model is correct; please see this elucidating blog post:

https://thestatsgeek.com/2014/01/25/r-squared-and-goodness-of-fit-in-linear-regression/

Note that opinions differ about the appropriate definition of r^2 when a model contains no intercept term. In R the definition of r^2 uses 0 in place of $\bar{y}$ in this circumstance, which has the somewhat undesirable effect of making the r^2 increase substantially if an intercept term is dropped from a model.

5.6 Prediction

Suppose that we have a fitted model, and would like to use the model to make predictions for a set of predictor variables values (which may have been used for fitting, or may be new). For example, suppose that a prediction is required for `tens` = 200 and `hard` = 50 using the model fitted to the `Rubber` data. The procedure is as follows.

1. Use the new predictor variables values to create a vector $\mathbf{x}_0$ in exactly the same way as the predictors would have been used to create a row of the design matrix if they had been used for model fitting.
2. The *prediction* is then $\hat{\mu}_0 = \mathbf{x}_0^\mathsf{T}\hat{\boldsymbol{\beta}}$.
3. Note that $\hat{\mu}_0$ is an estimator for the (conditional) mean $\mathbb{E}(y_0) = \mathbf{x}_0^\mathsf{T}\boldsymbol{\beta} = \mu_0$.

Obviously several predictions can be made simultaneously, for different predictor combinations, by replacing $\mathbf{x}_0$ by a prediction matrix, $\mathbf{X}_0$, say.

For example, the finally selected tyre wear model (by backward selection) was:

$$y_i = \beta_0 + \beta_1 t_i + \beta_2 t_i h_i + \beta_3 t_i^2 + \beta_4 h_i^2 t_i + \beta_5 t_i^3 + \beta_6 h_i^3 + \epsilon_i.$$

Hence to make a prediction for (h_0, t_0), we form the vector $\mathbf{x}_0^\mathsf{T} = [1, t_0, t_0 h_0, t_0^2, h_0^2 t_0, t_0^3, h_0^3]$, and the prediction is $\hat{\mu}_0 = \mathbf{x}_0^\mathsf{T}\hat{\boldsymbol{\beta}}$. With the values $h_0 = 50$ and $t_0 = 200$ then $\mathbf{x}_0^\mathsf{T} = [1, 200, 10000, 40000, 500000, 8000000, 125000]$. Let's calculate the corresponding prediction $\hat{\mu}_0$ in R.

```
x0 <- c(1, 200, 10000, 40000, 500000, 8000000, 125000)
t(x0)%*%coef(m4)

##           [,1]
## [1,] 236.0129
```

The R function `predict` automates the calculation of predictions. Notice how a *data frame*, `newdata`, is supplied containing the values at which predictions are required.

```
predict(m4, newdata = data.frame(tens = 200, hard = 50))

##        1
## 236.0129
```

This predicted value represents the abrasion loss of a tyre with an hardness of 50 and a tensile strength of 200.

Of course, we need more than just a point estimate to make informed choices/decisions. If the confidence interval for μ_0 is wide, we need to allow for outcomes far from the point estimate. Let's then determine the confidence interval for μ_0. We start by noting that

$$\mathbb{E}(\hat{\mu}_0) = \mathbb{E}(\mathbf{x}_0^\mathsf{T}\hat{\boldsymbol{\beta}}) = \mathbf{x}_0^\mathsf{T}\mathbb{E}(\hat{\boldsymbol{\beta}}) = \mathbf{x}_0^\mathsf{T}\boldsymbol{\beta} = \mu_0.$$

The variance of the prediction is also easily obtained using standard results on the transformation of covariance matrices and the covariance matrix of $\hat{\boldsymbol{\beta}}$, i.e.,

$$\begin{aligned}\text{var}(\hat{\mu}_0) = \text{var}(\mathbf{x}_0^\mathsf{T}\hat{\boldsymbol{\beta}}) &= \mathbf{x}_0^\mathsf{T}\text{var}(\hat{\boldsymbol{\beta}})(\mathbf{x}_0^\mathsf{T})^\mathsf{T} \\ &= \mathbf{x}_0^\mathsf{T}\mathbf{V}_{\hat{\boldsymbol{\beta}}}\mathbf{x}_0, \quad \text{where} \quad \mathbf{V}_{\hat{\boldsymbol{\beta}}} = \mathbf{R}^{-1}\mathbf{R}^{-T}\sigma^2 = (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\sigma^2.\end{aligned}$$

Also, because $\hat{\boldsymbol{\beta}}$ is normally distributed we have that

$$\hat{\mu}_0 \sim N(\mu_0, \sigma^2_{\hat{\mu}_0}), \quad \text{with} \quad \sigma^2_{\hat{\mu}_0} = \mathbf{x}_0^\mathsf{T}\mathbf{V}_{\hat{\boldsymbol{\beta}}}\mathbf{x}_0.$$

Standardising yields

$$\frac{\hat{\mu}_0 - \mu_0}{\sigma_{\hat{\mu}_0}} \sim N(0,1).$$

If we replace, $\sigma_{\hat{\mu}_0}$ by $\hat{\sigma}_{\hat{\mu}_0}$, which is equivalent to replace $\mathbf{V}_{\hat{\boldsymbol{\beta}}}$ by $\hat{\mathbf{V}}_{\hat{\boldsymbol{\beta}}}$ which, in turn, amounts to replace σ^2 by $\hat{\sigma}^2$ in $\mathbf{V}_{\hat{\boldsymbol{\beta}}}$, the resulting expression follows a t distribution with $n-p$ degrees of freedom. So an $100(1-\alpha)\%$ CI for μ_0 is

$$\hat{\mu}_0 \pm t_{1-\alpha/2,n-p}\hat{\sigma}_{\hat{\mu}_0},$$

where $t_{1-\alpha/2,n-p}$ is the value below which a t_{n-p} random variable will lie with probability $1-\alpha/2$.

The `predict` function also returns the estimated standard error of the prediction $\hat{\sigma}_{\hat{\mu}_0}$, as well as, a confidence interval for the expected response μ_0.

```
predict(m4, newdata = data.frame(tens = 200, hard = 50), se = TRUE)

## $fit
##        1
## 236.0129
##
## $se.fit
## [1] 16.64521
##
## $df
## [1] 23
##
## $residual.scale
## [1] 27.67362
```

Note that the last value, the residuals scale, is simply $\hat{\sigma}$, where $\hat{\sigma}^2 = \|\mathbf{r}\|^2/(n-p)$.

```
sigma(m4)

## [1] 27.67362

sqrt(sum(m4$residuals^2)/(nrow(Rubber) - length(coef(m4))))

## [1] 27.67362
```

Asking for the confidence interval we get the following.

```
predict(m4, newdata = data.frame(tens = 200, hard = 50), interval = "confidence")

##        fit      lwr      upr
## 1 236.0129 201.5797 270.4461
```

Note that the default confidence level is .95, but this can be altered using the `level` argument. Of course it is also possible to obtain several predictions (and associated standard errors and intervals) at once. For example

```
predict(m4, newdata = data.frame(tens = c(200, 190), hard = c(50, 60)),
        interval = "confidence")

##        fit      lwr      upr
## 1 236.0129 201.5797 270.4461
## 2 198.3464 176.7126 219.9802
```

Prediction intervals

The *confidence intervals* for predictions, considered so far, are random intervals that have some specified probability of including the true *expected response* corresponding to some predictor variable values. Sometimes it is useful to obtain **prediction intervals** which are intervals within which we would expect a new independent observation of a response variable, say y_0, to lie, with some specified probability, given some particular values for the predictors, say $\mathbf{x}_0$. In relation to the tyre wear example, this means we would be interested in knowing the range of reasonable abrasion loss values for a tyre with some given characteristics (i.e., for a specific hardness and tensile strength, e.g., 50 and 200, respectively). For the purpose of determining a prediction interval for a new observation y_0, we look at the prediction error $\hat{\epsilon}_0 = y_0 - \mathbf{x}_0^{\mathsf{T}}\hat{\beta} = y_0 - \hat{\mu}_0$. We have that

$$\mathbb{E}(\hat{\epsilon}_0) = \mathbb{E}(y_0 - \hat{\mu}_0) = \mathbb{E}(y_0) - \mathbb{E}(\hat{\mu}_0) = \mu_0 - \mu_0 = 0,$$

and

$$\text{var}(\hat{\epsilon}_0) = \text{var}(y_0) + \text{var}(\hat{\mu}_0) = \sigma^2 + \sigma^2_{\hat{\mu}_0}.$$

When calculating the above variance we have made use of the following facts: (1) the independence of y_0 and $\hat{\mu}_0$ (y_0 is a new observation, not used in the estimation of the model used to calculate $\hat{\mu}_0$), and (2) y_0 follows the same data generating mechanism as the other observations and as such its variance is σ^2. Because the difference of two normal random variables is also a normal random variable, it follows that

$$\hat{\epsilon}_0 \sim N(0, \sigma^2 + \sigma^2_{\hat{\mu}_0}).$$

Standardising and replacing $\hat{\sigma}^2$ for σ^2 yields

$$\frac{y_0 - \hat{\mu}_0}{(\hat{\sigma}^2 + \hat{\sigma}^2_{\hat{\mu}_0})^{1/2}} \sim t_{n-p},$$

and therefore

$$\begin{aligned} \Pr\left(-t_{1-\alpha/2,n-p} < \frac{y_0 - \hat{\mu}_0}{(\hat{\sigma}^2 + \hat{\sigma}^2_{\hat{\mu}_0})^{1/2}} < t_{1-\alpha/2,n-p}\right) &= 1-\alpha \\ \Rightarrow \Pr\left(\hat{\mu}_0 - t_{1-\alpha/2,n-p}(\hat{\sigma}^2 + \hat{\sigma}^2_{\hat{\mu}_0})^{1/2} < y_0 < \hat{\mu}_0 + t_{1-\alpha/2,n-p}(\hat{\sigma}^2 + \hat{\sigma}^2_{\hat{\mu}_0})^{1/2}\right) &= 1-\alpha \end{aligned}$$

i.e. a $100(1-\alpha)\%$ *prediction interval* for a new response observation at a given set of predictor values is

$$\hat{\mu}_0 \pm t_{1-\alpha/2,n-p}(\hat{\sigma}^2 + \hat{\sigma}^2_{\hat{\mu}_0})^{1/2}.$$

Per construction, the prediction interval is always wider than the corresponding confidence interval for μ_0. In applications with high error variance, the interval can be considerably wider.

Even though at first glance the two intervals appear to be very similar, they are indeed very different. In the first case, we constructed a confidence interval for $\mathbb{E}(y_0) = \mu_0$. This implies that the random interval overlaps the (fixed) mean $\mathbb{E}(y_0)$ with probability $1 - \alpha$. In the second case, we rather constructed an interval which is very likely (more precisely with probability $1 - \alpha$) to contain the (random) new observation.

Predictions intervals are easily obtained in R using the `predict` function.

```
predict(m4, newdata = data.frame(tens = 200, hard = 50), interval = "prediction")

##        fit     lwr      upr
## 1 236.0129 169.208 302.8178
```

Notice that this is much wider than the corresponding confidence interval for the predicted mean response. And it has to be, as this is the interval within which 95% of new replicate observations should lie (while the confidence interval was concerned with bracketing the long term average of such replicate observations).

6 Practical modelling with factors and interactions

Most of the models covered so far have been for situations in which the predictor variables are continuous variables (e.g., the speed in the `cars` dataset or the hardness and tensile strength in the `Rubber` dataset), but there are many situations in which the predictor variables are more qualitative in nature, and serve to divide the responses into groups. Examples might be eye-colour of subjects in a psychology experiment, which of three alternative hospitals were attended by patients in a drug trial, manufacturers of cars used in crash tests etc. Variables like these, which serve to classify the units on which the response variable has been measured into distinct categories, are known as *factor variables*, or simply *factors*. The different categories of the factor are known as *levels* of the factor. For example, levels of the factor 'eye colour' might be 'blue', 'brown', 'grey' and 'green', so that we would refer to eye colour as a factor with four levels. Note that 'levels' is quite confusing terminology: there is not necessarily any natural ordering of the levels of a factor. Hence 'levels' of a factor might best be thought of as 'categories' of a factor, or 'groups' of a factor.

Factor variables are handled using dummy(or indicator) variables. Each factor variable can be replaced by as many dummy variables as there are levels of the factor — one for each level of the factor. For each response datum, only one of these dummy variables will be non-zero: the dummy variable for the single level that applies to that response. Consider an example to see how this works in practice: 9 laboratory rats are fed too much, so that they divide into 3 groups of 3: 'fat', 'very fat' and 'enormous'. Their blood insulin levels are then measured 10 minutes after being fed a standard amount of sugar. The investigators are interested in the relationship between insulin levels and the factor 'rat size'. Hence a model could be set up in which the predictor variable is the factor 'rat size', with the three levels 'fat', 'very fat' and 'enormous'. Writing y_i for the i^{th} insulin level measurement, a suitable model might be:

$$\mathbb{E}(Y_i) \equiv \mu_i = \begin{cases} \beta_0 & \text{if rat is fat,} \\ \beta_1 & \text{if rat is very fat,} \\ \beta_2 & \text{if rat is enormous,} \end{cases}$$

and this is easily written in linear model form, using a dummy predictor variable for each level of the factor:

$$\begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \\ \mu_5 \\ \mu_6 \\ \mu_7 \\ \mu_8 \\ \mu_9 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{pmatrix}.$$

A key difference between dummy variables and directly measured predictor variables, is that the dummy variables, and parameters associated with a factor, are almost always treated as a group during model selection — it does not usually make sense for a subset of the dummy variables associated with a factor to be dropped or included on their own: either all are included or none. The F ratio tests derived in section 4.3.2 are designed for hypothesis testing in this situation.

6.1 Identifiability

When modelling with factor variables, model 'identifiability' becomes an important issue. In a linear model, the parameters are identifiable if whenever the parameters take different values, the expected value vector generated by the model also changes. If the design matrix $\mathbf{X}$ is full column rank, then the parameters in the linear model are identifiable. It is quite easy to set up models, involving factors, in which it is impossible to estimate the parameters uniquely, because an infinite set of alternative parameter vectors would give rise to exactly the same expected value vector. A simple example illustrates the problem. Consider again the fat rat example, but suppose that we wanted to formulate the model in terms of an overall mean insulin level, α, and deviations from that level, β_j, associated with each level of the factor. The model would be something like:

$$\mu_i = \alpha + \beta_j, \quad \text{if rat } i \text{ is rat size level } j, \tag{14}$$

where j is 0, 1 or 2, corresponding to 'fat', 'very fat' or 'enormous'. The problem with this model is that there is not a one-to-one correspondence between the parameters and the fitted values, so that the parameters can not be uniquely estimated from the data. This is easy to see. Consider any particular set of α and $\boldsymbol{\beta}$ values, giving rise to a particular $\boldsymbol{\mu}$ value: any constant c could be added to α and simultaneously subtracted from each element of $\boldsymbol{\beta}$ without changing the value of $\boldsymbol{\mu}$ (e.g., $\mu_1 = \alpha + \beta_0 = (\alpha + c) + (\beta_0 - c)$ and the same for $\mu_2, \ldots, \mu_9$). Hence there is an infinite set of parameters giving rise to each $\boldsymbol{\mu}$ value, and therefore the parameters can not be estimated uniquely. The model is not 'identifiable'.

This situation can be diagnosed directly from the model design matrix. Written out in full, the example model is

$$\begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \\ \mu_5 \\ \mu_6 \\ \mu_7 \\ \mu_8 \\ \mu_9 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta_0 \\ \beta_1 \\ \beta_2 \end{pmatrix}.$$

But the columns of the model matrix are not independent (as the first column is the sum of the second, third, and fourth columns), and this lack of full column rank means that the formulae for finding the least squares parameter estimates break down (in terms of section 4, $\mathbf{R}$ will not be full rank, and will hence not be invertible, or equivalently, $\mathbf{X}^\mathsf{T}\mathbf{X}$ will not be invertible). Identifiable models have model matrices of full column rank; unidentifiable ones are column rank deficient.

One possible solution to the identifiability problem is to set one of the unidentifiable parameters to zero, which requires only that the model is re-written, without the zeroed parameter. For example, in the fat rat case, we could set α to zero, and recover the original identifiable model we started with (however, as we shall see shortly, this will not work well if there is more than one factor as more than one parameter needs to be dropped). This is perfectly legitimate, since the reduced model is capable of reproducing any expected values that the original model could produce. When you specify models involving factors in R, it will automatically impose an identifiability constraint for you, and by default, and differently to what we just mentioned, the constraint will be that the parameter for the 'first' level of the factor is zero (this is a more general solution that can be easily extended to the case of multiple factor variables). Let us use dummy variables and write down the model in this case. To this end, let $x_{1i} = 1$ if the rat is very fat and $x_{1i} = 0$ otherwise and, similarly, $x_{2i} = 1$ if the rat is enormous and $x_{2i} = 0$ otherwise. Then

$$\mu_i = \alpha + \beta_1 x_{1i} + \beta_2 x_{2i}.$$

For fat rats, since $x_{1i} = x_{2i} = 0$, we have that $\mu_i = \alpha$. That is, in this parametrization, α represents the expected insulin levels for a fat rat and so its interpretation is different from the one in model (14). For very fat rats, $x_{1i} = 1$ and $x_{2i} = 0$, so $\mu_i = \alpha + \beta_1$, and for enormous rats, $x_{1i} = 0$ and $x_{2i} = 1$ and hence $\mu_i = \alpha + \beta_2$. Suppose that rat i is fat and rat i' is very fat, then $\mu_{i'} - \mu_i = \alpha + \beta_1 - \alpha = \beta_1$ and so this parameter represents the expected difference in insulin levels between a very fat and a fat rat. Similarly, β_2 represents the expected difference in insulin levels between an enormous rat and a fat rat. Because all comparisons are made to group of fat rats, this category is said to be the baseline or reference level. Further note that if rat i' is very fat and rat i'' is enormous then $\mu_{i'} - \mu_{i''} = (\alpha + \beta_1) - (\alpha + \beta_2) = \beta_1 - \beta_2$, i.e., the expected difference in insulin levels for a very fat rat and an enormous rat is $\beta_1 - \beta_2$. All possible choices of $c - 1$ indicators for c categories are equivalent, having the same model fit and the same results for inferences about the population means. The interpretations of the coefficients depend on the choice of reference category, but the differences between any two parameters used to compare groups are identical for each possible choice. Note, however, that the reference/baseline level should not be sparse. For the running example, suppose that there are very few fat rats (and that they are the baseline level) in the dataset. Then, for most cases, either one of x_1 or x_2 is equal to one and then $x_1 + x_2 = 1$ for all these (many) cases, implying near linear dependency between the columns of the design matrix (remember that the first column is a column of ones corresponding to the intercept). Although $\hat{\boldsymbol{\beta}}$ is computable (i.e., we can invert $\mathbf{R}$ or $(\mathbf{X}^\mathsf{T}\mathbf{X})$) it would be numerically unstable.

In the one factor case, this discussion of identifiability may seem to be un-necessarily complicated, since it is so easy to write down the model directly, in an identifiable form. However, when models involve more than one factor variable, the issue can not be avoided.

6.2 Multiple factors

It is frequently the case that more than one factor variable should be included in a model, and this is straightforward to do. Continuing the fat rat example, it might be that the sex of the rats is also a factor in insulin production, and that the appropriate model is:

$$\mu_i = \alpha + \beta_j + \gamma_k \text{ if rat } i \text{ is rat size level } j \text{ and sex } k$$

where k is 0 or 1 for male or female. Written out in full (assuming the rats are MMFFFMFMM) the model is

$$\begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \\ \mu_5 \\ \mu_6 \\ \mu_7 \\ \mu_8 \\ \mu_9 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta_0 \\ \beta_1 \\ \beta_2 \\ \gamma_0 \\ \gamma_1 \end{pmatrix}.$$

It is immediately obvious that the model matrix is of column rank 4, implying that two constraints are required to make the model identifiable. You can see the lack of column independence by noting that column 5 is column 1 minus column 6, while column 2 is column 1 minus columns 3 and 4. An obvious pair of constraints would be to set $\beta_0 = \gamma_0 = 0$ (and this aligns with what `R` does: exclude the first level of each factor variable), so that the full model is

$$\begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \\ \mu_5 \\ \mu_6 \\ \mu_7 \\ \mu_8 \\ \mu_9 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta_1 \\ \beta_2 \\ \gamma_1 \end{pmatrix}.$$

Note that in this last parametrization the intercept α represents the expected insulin levels for a fat male rat.

6.3 'Interactions' of factors

In the examples considered so far, the effect of factor variables has been purely additive, but it is possible that a response variable may react differently to the combination of two factors, than would be predicted by simply adding the effect of the two factors separately. For example, if examining patient blood cholesterol levels, we might consider the factors 'hypertensive' (yes/no) and 'diabetic' (yes/no). Being hypertensive or diabetic would be expected to raise cholesterol levels, but being both is likely to raise cholesterol levels much more than would be predicted from just adding up the apparent effects when only one condition is present. When the effect of two factor variables together differs from the sum of their separate effects, then they are said to *interact*, and an adequate model in such situations requires *interaction terms*. Put another way, if the effects of one factor change in response to another factor, then the factors are said to interact.

Let us continue with the fat rat example, but now suppose that how insulin level depends on size varies with sex. An appropriate model is then

$$\mu_i = \alpha + \beta_j + \gamma_k + \delta_{jk} \text{ if rat } i \text{ is rat size level } j \text{ and sex } k,$$

where the δ_{jk} terms are the parameters for the interaction of rat size and sex. Writing this model out in full it is clear that it is spectacularly unidentifiable:

$$\begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \\ \mu_5 \\ \mu_6 \\ \mu_7 \\ \mu_8 \\ \mu_9 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta_0 \\ \beta_1 \\ \beta_2 \\ \gamma_0 \\ \gamma_1 \\ \delta_{00} \\ \delta_{01} \\ \delta_{10} \\ \delta_{11} \\ \delta_{20} \\ \delta_{21} \end{pmatrix}.$$

In fact, for this simple example, with rather few rats, we now have more parameters than data. There are of course many ways of constraining this model to achieve identifiability. One possibility (the default in R) is to set $\beta_0 = \gamma_0 = \delta_{00} = \delta_{01} = \delta_{10} = \delta_{20} = 0$. The resulting model can still produce any fitted value vector that the full model can produce, but all the columns of its model matrix are independent, so that the model is identifiable:

$$\begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \\ \mu_5 \\ \mu_6 \\ \mu_7 \\ \mu_8 \\ \mu_9 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta_1 \\ \beta_2 \\ \gamma_1 \\ \delta_{11} \\ \delta_{21} \end{pmatrix}.$$

Of course, the more factor variables are present, the more interactions are possible, and the higher the order of the possible interactions: for example if three factors are present then each factor could interact with each factor, giving three possible 'two-way' interactions, while all the factors could also interact together, giving a three-way interaction (e.g. the way in which insulin levels dependence on rat size is influenced by sex is itself influenced by blood group — perhaps with interactions beyond two-way, equations are clearer than words).

6.4 Factor continuous interactions

Suppose we had access to the exact weight, w_i, of the rats, rather than only its classification into one of the three groups previously defined. In this case

$$\mu_i = \alpha + \gamma_j + \beta w_i + \delta_j w_i \text{ if rat } i \text{ is sex } j$$

might be appropriate. That is to say insulin levels vary linearly with weight, but in a different way for male and female rats. Here γ_j would be the 'main effect' of sex, βw_i the 'main effect' of weight, and $\delta_j w_i$ is the weight-sex 'interaction'. Similar identifiability issues exist between the main effect of sex and the weight-sex interaction as exist between factor interactions and main effects, and similar constraints are needed. Recalling the assumed

ordering of rats, MMFFFMFMM, then an identifiable version of the model is

$$\begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \\ \mu_5 \\ \mu_6 \\ \mu_7 \\ \mu_8 \\ \mu_9 \end{pmatrix} = \begin{pmatrix} 1 & 0 & w_1 & 0 \\ 1 & 0 & w_2 & 0 \\ 1 & 1 & w_3 & w_3 \\ 1 & 1 & w_4 & w_4 \\ 1 & 1 & w_5 & w_5 \\ 1 & 0 & w_6 & 0 \\ 1 & 1 & w_7 & w_7 \\ 1 & 0 & w_8 & 0 \\ 1 & 0 & w_9 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \gamma_1 \\ \beta \\ \delta_1 \end{pmatrix}.$$

6.5 Using factor variables in R

It is very easy to work with factor variables in R. All that is required is that you let R know that a particular variable is a factor variable. For example, suppose z is a variable declared as follows:

```
z <- c(1, 1, 1, 2, 2, 1, 3, 3, 3, 3, 4)
z

##  [1] 1 1 1 2 2 1 3 3 3 3 4
```

and it is to be treated as a 4 level factor. The function `as.factor()` will ensure that `z` is treated as a factor:

```
z <- as.factor(z)
z

##  [1] 1 1 1 2 2 1 3 3 3 3 4
## Levels: 1 2 3 4
```

When a factor variable is printed, a list of its levels is also printed — this provides an easy way to tell if a variable is a factor variable. Note also that the digits of `z` are treated purely as labels: the numbers 1 to 4 could have been any labels. For example, `x` could be a factor with 3 levels, declared as follows:

```
x <- c("A","A","C","C","C","er","er")
x

## [1] "A"  "A"  "C"  "C"  "C"  "er" "er"

x <- as.factor(x)
#check reference level (the row that only contains zeroes)
contrasts(x)

##    C er
## A  0  0
## C  1  0
## er 0  1

#want to change the reference level from A to er? Use the relevel function
x <- relevel(x, ref = "er")
x

## [1] A  A  C  C  C  er er
## Levels: er A C

contrasts(x)

##    A C
## er 0 0
## A  1 0
## C  0 1
```

Once a variable is declared as a factor variable, then R can process it automatically within model formulae, by replacing it with the appropriate number of binary dummy variables (and imposing any necessary identifiability constraints on the specified model).

It is worth emphasizing that factor predictors should not be coded as continuous predictors. In general, as already mentioned at the starting of this section, there is not necessarily any natural ordering of the levels of a factor. To give a concrete example, let us use the vector x in the example given just above, which has levels A, C, and er. Suppose we would code x as a numerical predictor taking the values 1, 2, and 3, respectively for A, C, and er. This implicitly assumes an order in the values of x, since A is closer to C than to er. This also implicitly assumes that er is three times higher than A. Further, the coding is completely arbitrary – why not consider 1, 1.5, and 1.75 instead? The right way of dealing with categorical predictors in regression to set the variable as a factor and let R do the 'dummification' internally.

6.5.1 Factor interactions in model formulae

Within model formulae the notation a:b indicates that the interaction of variables a and b should be included in the model, while a*b is exactly equivalent to a + b + a:b, meaning that main effects and interactions should be included. Note that if a and b are both factors then y ~ a:b and y ~ a*b result in exactly the same model fits, but with different meanings for the parameters and different identifiability constraints.

With reference to the fat rat examples, if insulin is the response, size and sex are the factor variables, and weight the continuous variable, then

- insulin ~ size - 1 gives the section 6 model.
- insulin ~ size gives the section 6.1 model (same fit as above, but different parameter interpretation).
- insulin ~ size + sex gives the section 6.2 model.
- insulin ~ size*sex gives the section 6.3 model.
- insulin ~ sex*weight gives the section 6.4 model.

Another useful notation is provided by terms like (a+b+c)^2, which specifies main effects and interactions up to the order of the given power - 2 in this case. i.e. (a+b+c)^2 = a + b + c + a:b + a:c + b:c.

6.5.2 A simple example

As an example of the use of factor variables, consider the PlantGrowth data frame supplied with R. These are data on the growth of plants under control conditions and two different conditions. The factor group has three levels cntrl, trt1 and trt2, and it is believed that the growth of the plants depends on this factor. First check that group is already a factor variable:

```
PlantGrowth$group
```

```
##  [1] ctrl ctrl ctrl ctrl ctrl ctrl ctrl ctrl ctrl ctrl trt1 trt1 trt1 trt1 trt1
## [16] trt1 trt1 trt1 trt1 trt1 trt2 trt2 trt2 trt2 trt2 trt2 trt2 trt2 trt2 trt2
## Levels: ctrl trt1 trt2
```

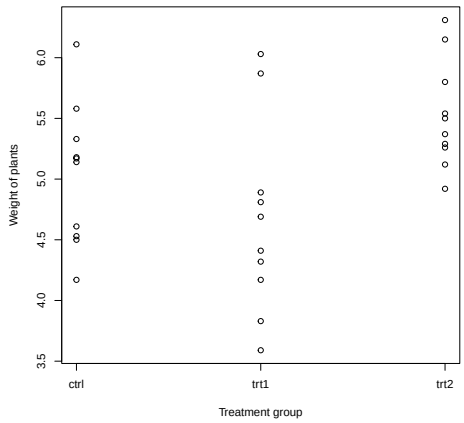
. . . since a list of levels is reported, it must be. But we can always double check.

```
is.factor(PlantGrowth$group)
```

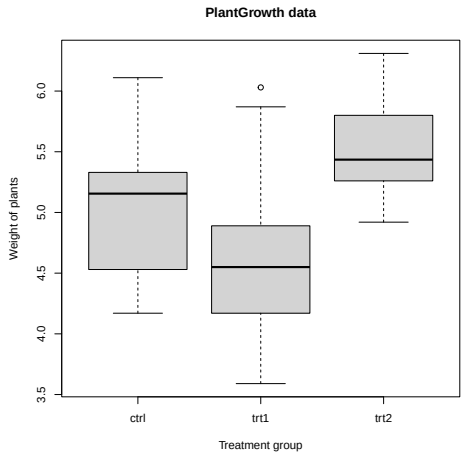
```
## [1] TRUE
```

The response variable for these data is weight of the plants at some set time after planting, and the aim is to investigate whether the group factor controls this, and if so, to what extent. We start by visualising the data, and below there are two common ways of doing it.

```r
stripchart(weight ~ group, data = PlantGrowth, vertical = TRUE, pch = 1,
           ylab = "Weight of plants", xlab = "Treatment group")
```

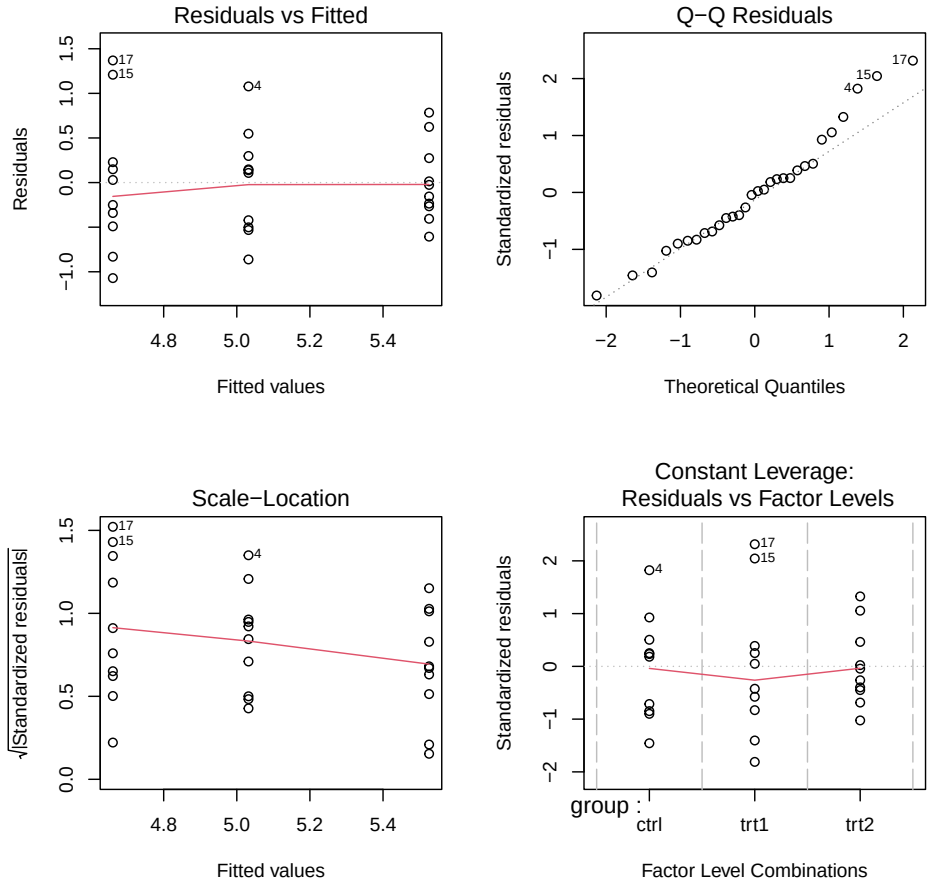



```r
boxplot(weight ~ group, data = PlantGrowth, main = "PlantGrowth data",
        ylab = "Weight of plants", xlab = "Treatment group")
```



Treatment 2 seems to stand out in terms of the corresponding weight of the plants. Let us fit the model now.

```r
pgm.1 <- lm(weight ~ group, data = PlantGrowth)

par(mfrow = c(2, 2))
plot(pgm.1)
```



As usual, the first thing to do, after fitting a model, is to check the residual plots. In this case there is some suggestion of decreasing variance with increasing mean, but the effect does not look very pronounced, so it is probably safe to proceed. Note that, since in this case the leverages are all the same for each observation, the lower right plot is now simplified (actually, in this specific example, it does not add any new information beyond what we already have in the other three plots).

```
summary(pgm.1)

##
## Call:
## lm(formula = weight ~ group, data = PlantGrowth)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.0710 -0.4180 -0.0060  0.2627  1.3690
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   5.0320     0.1971  25.527   <2e-16 ***
## grouptrt1    -0.3710     0.2788  -1.331   0.1944
## grouptrt2     0.4940     0.2788   1.772   0.0877 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6234 on 27 degrees of freedom
```

```
## Multiple R-squared:  0.2641,Adjusted R-squared:  0.2096
## F-statistic: 4.846 on 2 and 27 DF,  p-value: 0.01591

#check first rows of model matrix used in the fit
head(model.matrix(pgm.1))

##   (Intercept) grouptrt1 grouptrt2
## 1           1         0         0
## 2           1         0         0
## 3           1         0         0
## 4           1         0         0
## 5           1         0         0
## 6           1         0         0
```

Notice how R reports an intercept parameter and parameters for the two treatment levels, but, in order to obtain an identifiable model, it has not included a parameter for the control level of the group factor. So the estimated overall mean weight (in the population that these plants represent, given control conditions) is 5.032, while treatment 1 is estimated to lower this weight by 0.37, and treatment 2 to increase it by 0.49. However, neither parameter individually appears to be significantly different from zero.

Model selection based on the summary output is very difficult for models containing factors. It makes little sense to drop the dummy variable for just one level of a factor from a model, and if we did, what would we then do about the model identifiability constraints? Usually, it is only of interest to test whether the whole factor variable should be in the model or not, and this amounts to testing whether all its associated parameters are simultaneously zero or not. The F-ratio tests derived in section 4.3.2 are designed for just this purpose. For example, we would compare `pgm.1` to a model in which the expected response is given by a single parameter (an intercept!) that does not depend on `group` and this is exactly the hypothesis test associated to the F-statistic reported in the very last line of the summary output. The p-value is 0.016 suggesting that the null hypothesis that the group variable does not have an effect on weight is not very plausible. So we see that the data provide evidence for an effect of the `group` factor variable on `weight`, which appeared marginal or absent when we examined p-values for the individual model parameters. This comes about because we have, in effect, considered all the parameters associated with the factor simultaneously, thereby obtaining a more powerful test than any of the single parameter tests could be. In the light of this analysis, the most promising treatment to look at is clearly treatment 2, since this gives the largest and 'most significant' effect and it is a positive effect. Of course, the same could have been obtained through the use of the `anova` function.

```
pgm.0 <- lm(weight ~ 1, data = PlantGrowth)
anova(pgm.0, pgm.1)

## Analysis of Variance Table
##
## Model 1: weight ~ 1
## Model 2: weight ~ group
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1     29 14.258
## 2     27 10.492  2    3.7663 4.8461 0.01591 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Let us now create a new group variable and change the baseline level from the control group to the treatment 1 group.

```
PlantGrowth$group_alt <- relevel(PlantGrowth$group, ref = "trt1")
pgm.1_alt <-  lm(weight ~ group_alt, data = PlantGrowth)
summary(pgm.1_alt)

##
## Call:
## lm(formula = weight ~ group_alt, data = PlantGrowth)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.0710 -0.4180 -0.0060  0.2627  1.3690
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)     4.6610     0.1971  23.644  < 2e-16 ***
## group_altctrl   0.3710     0.2788   1.331  0.19439
## group_alttrt2   0.8650     0.2788   3.103  0.00446 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6234 on 27 degrees of freedom
## Multiple R-squared:  0.2641,Adjusted R-squared:  0.2096
## F-statistic: 4.846 on 2 and 27 DF,  p-value: 0.01591
```

Apart from the coefficients part, the rest of the output is exactly the same. Note that under the first model (in which the control group is the baseline) the estimated mean plant growth in the control group is 5.0320, in the treatment 1 group is $5.0320 - 0.3710 = 4.661$ and in the treatment group 2 is $5.0320 + 0.4940 = 5.526$. In turn, in the second model (in which the treatment 1 group is the baseline), the estimated mean plant growth in the control group is $4.6610 + 0.3710 = 5.032$, in the first treatment group is 4.6610 and in the second treatment group is $4.6610 + 0.8650 = 5.526$. Further, and for instance, under the first model, the estimated difference in mean plant growth between treatment group 1 and 2 is $-0.3710 - 0.4940 = -0.865$, whereas such difference in the second model is given by -0.8650.

6.6 The `warpbreak` data

Now consider an example with two factors as predictors. The `warpbreak` data in R come from an industrial experiment, attempting to find optimal conditions for weaving with the minimum number of breaks in the wool being woven. Two types of `wool` (A and B) were tested at each of 3 weaving `tension`s (**L**ow, **M**edium, **H**igh), with 9 replicates per each combination of wool type and tension level, and the number of `breaks` in a standard length of cloth was recorded. The experiment was designed in such a way that replicates for each combination of wool type and tension level are independent. Let's first have a quick look at the data.

```
names(warpbreaks)

## [1] "breaks"  "wool"    "tension"

warpbreaks$wool

##  [1] A A A A A A A A A A A A A A A A A A A A A A A A A A A B B B B B B B B B
## [39] B B B B B B B B B B B B B B B
## Levels: A B

warpbreaks$tension

##  [1] L L L L L L L L L M M M M M M M M M H H H H H H H H H L L L L L L L L M M
## [39] M M M M M M M M H H H H H H H H H
## Levels: L M H
```

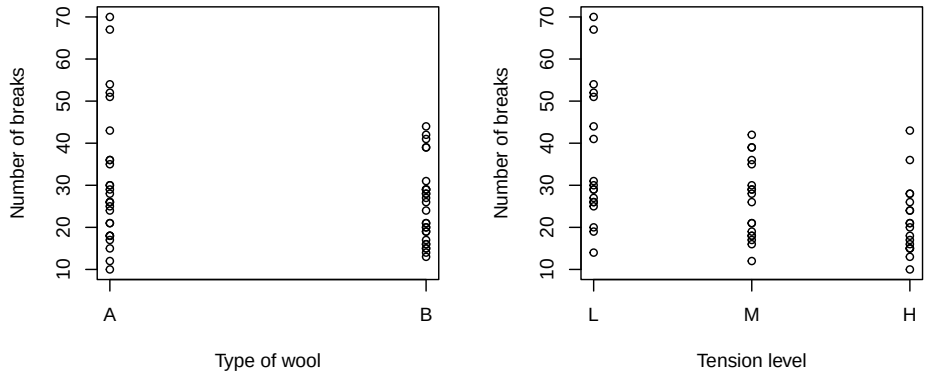
The following produce a pretty display of the data.

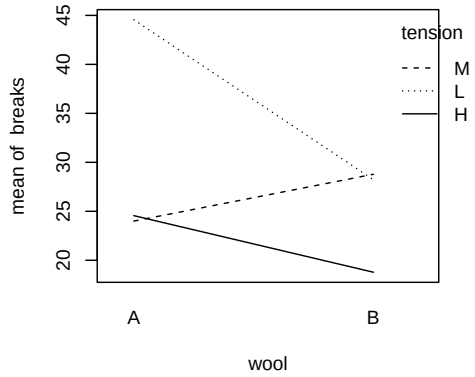
```
warpbreaks$replicate <- rep(1:9, len = 54)
ftable(xtabs(breaks ~ wool + tension + replicate, data = warpbreaks))

##             replicate  1  2  3  4  5  6  7  8  9
```

```
## wool tension
## A    L                 26 30 54 25 70 52 51 26 67
##      M                 18 21 29 17 12 18 35 30 36
##      H                 36 21 24 18 10 43 28 15 26
## B    L                 27 14 29 19 29 31 41 20 44
##      M                 42 26 19 16 39 28 21 39 29
##      H                 20 21 24 17 13 15 15 16 28
```

Simple plots of the data can be obtained as follows.

```
par(mfrow = c(2, 2))
stripchart(breaks ~ wool, data = warpbreaks, vertical = TRUE, pch = 1,
           ylab = "Number of breaks", xlab = "Type of wool")
stripchart(breaks ~ tension, data = warpbreaks, vertical = TRUE, pch = 1,
           ylab = "Number of breaks", xlab = "Tension level")
with(warpbreaks, interaction.plot(wool,tension,breaks))
```

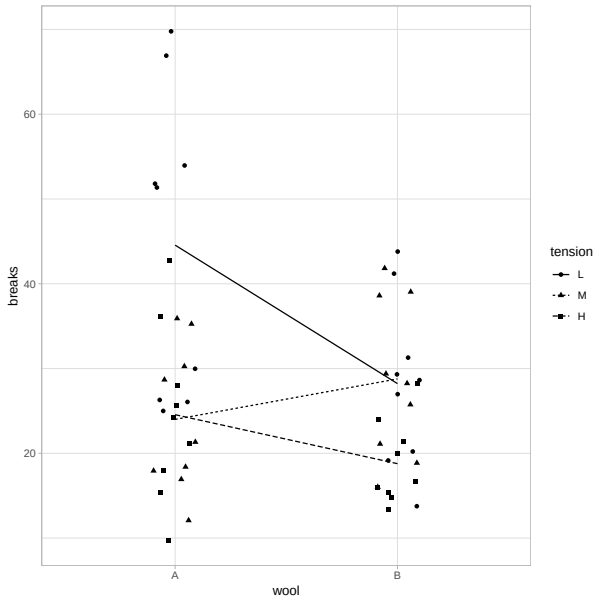




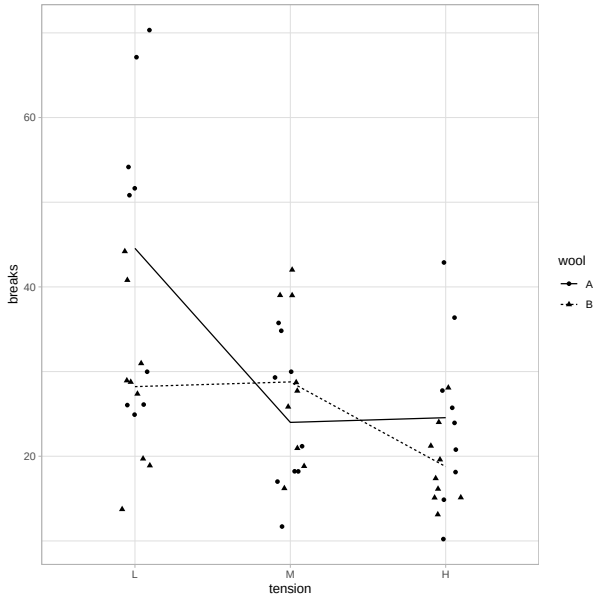
The plot in the second row should be repeated, reversing the roles of the two predictors. More informative but harder-to-construct plots of the data are as follows (code from Faraway, 2014, p. 243).

```
require(ggplot2)
ggplot(warpbreaks, aes(x = wool,y = breaks, shape = tension)) +
  geom_point( position = position_jitter(width = .1)) +
  stat_summary(fun = "mean", geom = "line",
```

```
              aes(group = tension, linetype = tension)) +
  theme(legend.position = "top", legend.direction = "horizontal") +
  theme_light()
```




```
ggplot(warpbreaks, aes(x = tension, y = breaks,shape=wool)) +
  geom_point( position = position_jitter(width = .1)) +
  stat_summary(fun = "mean", geom = "line",
               aes(group=wool, linetype = wool)) +
  theme(legend.position = "top", legend.direction = "horizontal") +
  theme_light()
```



There is some evidence of non-constant variation. We can also observe that there may be some interaction between the factors as the lines joining the mean response at each level are not close to parallel. Let us therefore consider a model with an interaction between wool type and tension level. Specifically, letting y_{ijk} to denote the number of

breaks for the k^{th} replicate at the i^{th} wool type and j^{th} tension, a possible model is

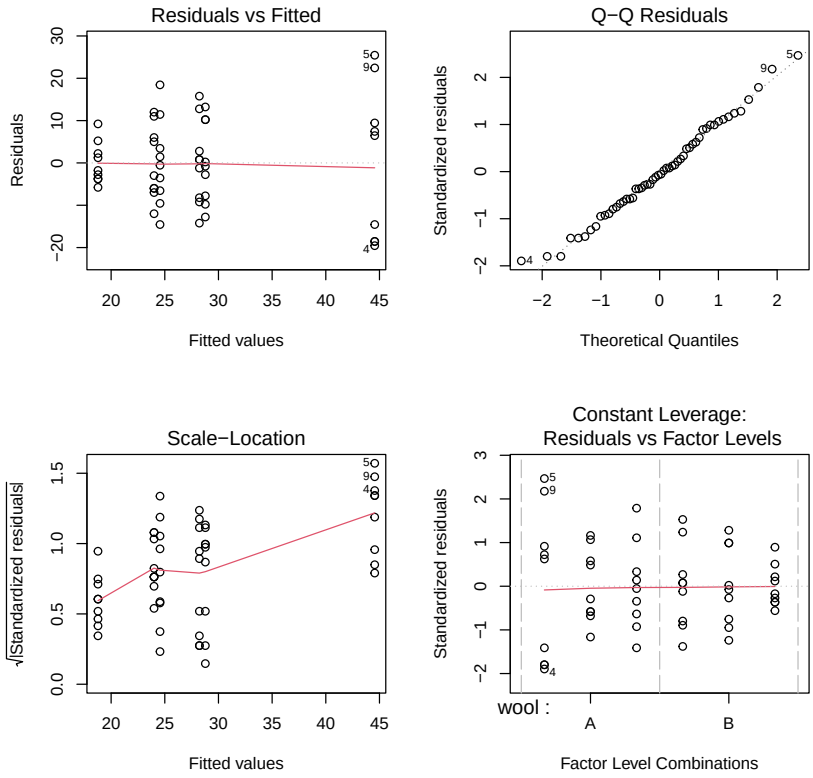
$$y_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon_{ijk}, \quad i \in \{A, B\}, \quad j \in \{L, M, H\}, \quad k \in \{1, \ldots, 9\}, \tag{15}$$

where the 2 α_i parameters are for the *main effect* of wool, the 3 β_j parameters are for the main effect of tension and the 6 γ_{ij} parameters represent the interaction of wool type and tension (μ is the overall expected number of breaks). Identifiability constraints are needed for this model, of course, but these are generated automatically in R. We already know what R does by default: set α_A, β_L, γ_{AL}, γ_{AM}, γ_{AH}, and γ_{BL} to zero. Note that the model actually only has six identifiable parameters (rather than the 12 originally specified). Let's the fit the model.

```
wm0 <- lm(breaks ~ wool*tension, data = warpbreaks)
#equivalent to wm0 <-lm(breaks ~ wool+ tension + wool:tension, data = warpbreaks)
```

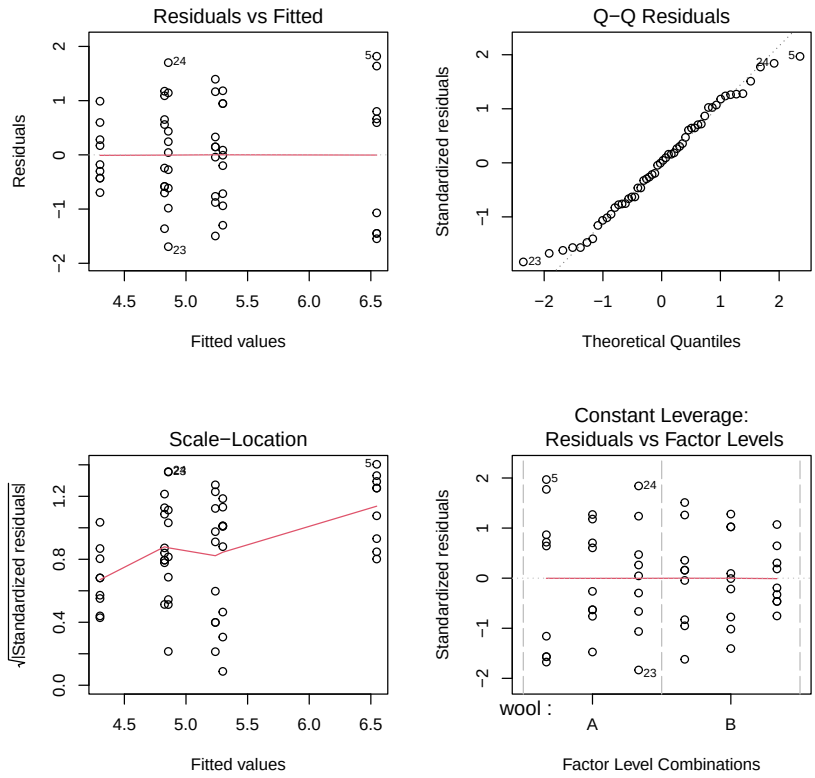
As always, let us check the model fit.

```
par(mfrow = c(2, 2))
plot(wm0)
```



The variance seems to be increasing with the mean here (left 2 panels of the residual plots). This is perhaps not surprising as the data are really counts of fairly rare events: something that might be best modelled by a Poisson distribution, rather than a normal. Transformation of the response data (the y's — `break`) can help in this circumstance. Power or log transformations are often worth trying. For Poisson data it can be shown that a square root power transform is sensible, so it is worth trying that.

```
wm <- lm(sqrt(breaks) ~ wool*tension, data = warpbreaks)
par(mfrow = c(2, 2))
plot(wm)
```



The revised plots show modest improvement, and experimenting with other transformations does not do much better, so the square root transform is probably sensible. Next, have a quick look at the model summary.

```
summary(wm)

##
## Call:
## lm(formula = sqrt(breaks) ~ wool * tension, data = warpbreaks)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.69410 -0.70129  0.01772  0.66046  1.81902
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   6.5476     0.3266  20.046  < 2e-16 ***
## woolB        -1.3094     0.4619  -2.835 0.006692 **
## tensionM     -1.7216     0.4619  -3.727 0.000511 ***
## tensionH     -1.6912     0.4619  -3.661 0.000625 ***
## woolB:tensionM  1.7821     0.6533   2.728 0.008874 **
## woolB:tensionH  0.7553     0.6533   1.156 0.253350
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9799 on 48 degrees of freedom
## Multiple R-squared:  0.3606,Adjusted R-squared:  0.294
## F-statistic: 5.415 on 5 and 48 DF,  p-value: 0.0004998
```

Note that we can write the two way ANOVA model in Equation (15) using dummy variables. To this end, we will

let l to index a combination of wool type, tension level, and replicate, for $l = 1, \ldots, 54$. That is, l would be the index of a row like those shown below.

```
head(warpbreaks)

##   breaks wool tension replicate
## 1     26    A       L         1
## 2     30    A       L         2
## 3     54    A       L         3
## 4     25    A       L         4
## 5     70    A       L         5
## 6     52    A       L         6
```

We thus write

$$y_l = \mu + \alpha_B \text{wood}_{Bl} + \beta_M \text{tension}_{Ml} + \beta_H \text{tension}_{Hl} + \gamma_{BM} \text{wood}_{Bl} \text{tension}_{Ml} + \gamma_{BH} \text{wood}_{Bl} \text{tension}_{Hl} + \epsilon_l$$

where $\text{wood}_{Bl} = 1$ if the wool type for observation l is B and 0 otherwise, $\text{tension}_{Ml} = 1$ if the tension level for observation l is M and 0 otherwise, and $\text{tension}_{Hl} = 1$ if the tension level for observation l is H and 0 otherwise. Note that for modelling the interaction effect we have to consider all possible combinations of the levels of wool type and tension (with the exception of the reference categories). The coefficients can be interpreted as follows:

- μ: expected number of breaks if wool type is A (reference) and tension level is L (reference).
- $\mu + \alpha_B$: expected number of breaks if wool type is B and tension level is L (reference).
- $\mu + \beta_M$: expected number of breaks if wool type is A (reference) and tension level is M.
- $\mu + \beta_H$: expected number of breaks if wool type is A (reference) and tension level is H.
- $\mu + \alpha_B + \beta_M + \gamma_{BM}$: expected number of breaks if wool type is B and tension level is M.
- $\mu + \alpha_B + \beta_H + \gamma_{BH}$: expected number of breaks if wool type is B and tension level is H.

The estimates of each of the parameters are provided in the above summary output. It is worth mentioning that γ_{BH} in the last line is an adjustment that is added on when wool type is B and tension is H, i.e., it is how much different the expected number of breaks is to what one would expect if the effects of wool type B and tension level H simply added to each other. An analogous interpretation holds for γ_{BM}.

Notice also from the summary output of the fitted model that the model actually only explains some 30% of the variability in the data (adjusted R-squared). As always with factor variables, it is not sensible to try and base model selection on the p-values in the summary. To perform model selection we can test the null hypothesis that the data are actually described by the model

$$y_l = \mu + \alpha_B \text{wood}_{Bl} + \beta_M \text{tension}_{Ml} + \beta_H \text{tension}_{Hl} + \epsilon_l,$$

(or in the typical notation of a two way ANOVA model: $y_{ijk} = \mu + \alpha_i + \beta_j + \epsilon_{ijk}$), against the alternative that the model with interactions is required. As already alluded to several times, an F-ratio test is the way to do this, as follows.

```
wm1 <- lm(sqrt(breaks) ~ wool + tension, data = warpbreaks)
anova(wm1, wm)

## Analysis of Variance Table
##
## Model 1: sqrt(breaks) ~ wool + tension
## Model 2: sqrt(breaks) ~ wool * tension
##   Res.Df    RSS Df Sum of Sq    F  Pr(>F)
## 1     50 53.291
## 2     48 46.089  2    7.2014 3.75 0.03067 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The p-value is quite low here, suggesting that there is evidence to reject the null hypothesis, and accept the alternative that the model with interaction is required (recall the reasoning: the p-value indicates that the observed F ratio is rather improbable under the null, suggesting that the null is not true).

Note that it is meaningless to try and remove main effects while leaving in a corresponding interaction, a point which will be elaborated shortly (in general, we neither test nor interpret the main effects of predictors that interact). For the moment this means that we are done, and have selected a model (the original one). So, the way that tension influences number of breaks is dependent on type of wool.

6.6.1 Follow up

Having selected the model, which says that the number of breaks depends on an interaction of wool type and tension, we would like to know the combination(s) of wool type and tension that minimize breakages. From the summary of the `wm` fitted model object this is not all that easy to work out, because of the way that the identifiability constraints have changed the meanings of the remaining parameters (also, see the interpretation of the coefficients provided just above). However, we can modify the identifiability constraints in order to get more interpretable parameters. To this end, it is easier to define a new factor variable, say WT, whose number of levels equals the product of the two factor levels. Let

$$WT = \begin{cases} AL, \text{ if wool type is A and tension is L,} \\ AM, \text{ if wool type is A and tension is M,} \\ AH, \text{ if wool type is A and tension is H,} \\ BL, \text{ if wool type is B and tension is L,} \\ BM, \text{ if wool type is B and tension is M,} \\ BH, \text{ if wool type is B and tension is H.} \end{cases}$$

We know that there are only six identifiable parameters, and so we will let the intercept of the model to be zero. Using dummy variables, the model can then be written as

$$y_l = \gamma_{AL} WT_{AL,l} + \gamma_{AM} WT_{AM,l} + \gamma_{AH} WT_{AH,l} + \gamma_{BL} WT_{BL,l} + \gamma_{BM} WT_{BM,l} + \gamma_{BH} WT_{BH,l} + \epsilon_l,$$

where $WT_{AL,l} = 1$ if observation l is at level AL of variable WT (i.e., if for observation l wool type is A and tension level is L) and 0 otherwise. The other dummy variables $WT_{AM,l}$, $WT_{AH,l}$, $WT_{BL,l}$, $WT_{BM,l}$, and $WT_{BH,l}$ are defined in a similar fashion. Under this formulation, γ_{AL} is the expected number of breaks for wool type A and tension L, γ_{AM} is the expected number of breaks for wool type A and tension M, etc: just what is needed. We can achieve this in `R` as follows.

```
wma <- lm(sqrt(breaks) ~ wool:tension - 1, data = warpbreaks)

#check the range of fitted values for this model is the same as for model wm.
range(fitted(wm) - fitted(wma))

## [1] -7.105427e-15  5.329071e-15

summary(wma)

##
## Call:
## lm(formula = sqrt(breaks) ~ wool:tension - 1, data = warpbreaks)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.69410 -0.70129  0.01772  0.66046  1.81902
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
```

```
## woolA:tensionL   6.5476     0.3266   20.05   <2e-16 ***
## woolB:tensionL   5.2381     0.3266   16.04   <2e-16 ***
## woolA:tensionM   4.8259     0.3266   14.78   <2e-16 ***
## woolB:tensionM   5.2987     0.3266   16.22   <2e-16 ***
## woolA:tensionH   4.8564     0.3266   14.87   <2e-16 ***
## woolB:tensionH   4.3022     0.3266   13.17   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9799 on 48 degrees of freedom
## Multiple R-squared:  0.9697,Adjusted R-squared:  0.9659
## F-statistic: 255.8 on 6 and 48 DF,  p-value: < 2.2e-16
```

Note the important point that this is not a different model to `wm`: just a different parameterization of the same model. The R squared and adjusted R squared are different from those of `wm` model because this model does not contain an intercept (and we know that this leads to an inflation of these quantities). We can immediately see that wool type B at high tension is estimated to give the lowest break rate. But given the uncertainty can we really be sure that this combination is better than wool A at medium or high tension? Such questions may be important if there are other considerations to take into account when choosing between wool types and tension levels (e.g. cost, wear rate of machinery, etc).

One obvious thing to do would be to find a confidence interval for the difference in break rate between wool B at high tension and wool A at medium tension. i.e. a confidence interval for $\gamma_{AM} - \gamma_{BH}$. To do this we need to find the standard deviation of $\hat{\gamma}_{AM} - \hat{\gamma}_{BH}$. We know that $\widehat{\text{var}}(\hat{\gamma}_{AM} - \hat{\gamma}_{BH}) = \widehat{\text{var}}(\hat{\gamma}_{AM}) + \widehat{\text{var}}(\hat{\gamma}_{BH}) - 2\widehat{\text{cov}}(\hat{\gamma}_{AM}, \hat{\gamma}_{BH})$. From the summary output above we have that $\widehat{\text{var}}(\hat{\gamma}_{AM}) = \widehat{\text{var}}(\hat{\gamma}_{BH}) = 0.3266^2 = 0.106688$. The estimated covariance matrix for the $\hat{\gamma}$'s parameters is also easily obtained.

```
vcov(wma)
```

```
##                 woolA:tensionL woolB:tensionL woolA:tensionM woolB:tensionM
## woolA:tensionL        0.106688       0.000000       0.000000       0.000000
## woolB:tensionL        0.000000       0.106688       0.000000       0.000000
## woolA:tensionM        0.000000       0.000000       0.106688       0.000000
## woolB:tensionM        0.000000       0.000000       0.000000       0.106688
## woolA:tensionH        0.000000       0.000000       0.000000       0.000000
## woolB:tensionH        0.000000       0.000000       0.000000       0.000000
##                 woolA:tensionH woolB:tensionH
## woolA:tensionL        0.000000       0.000000
## woolB:tensionL        0.000000       0.000000
## woolA:tensionM        0.000000       0.000000
## woolB:tensionM        0.000000       0.000000
## woolA:tensionH        0.106688       0.000000
## woolB:tensionH        0.000000       0.106688
```

It's diagonal: the parameter estimators are independent, thus implying that $\widehat{\text{cov}}(\hat{\gamma}_{AM}, \hat{\gamma}_{BH}) = 0$! Hence, $\widehat{\text{var}}(\hat{\gamma}_{AM} - \hat{\gamma}_{BH}) = 2 \times 0.106688 \approx 0.2134$. The required 95% confidence interval is therefore given by

$$(\hat{\gamma}_{AM} - \hat{\gamma}_{BH}) \pm t_{0.975,54-6} \times \sqrt{\widehat{\text{var}}(\hat{\gamma}_{AM} - \hat{\gamma}_{BH})} = (4.8259 - 4.3022) \pm 2.011 \times \sqrt{0.2134} = 0.52 \pm 0.93$$

Because the confidence interval contains the value zero, we really can't tell apart wool B at high tension and wool A at medium (or high) tension.

In this example it was easy to get R to use a parameterization that was convenient for extracting the main result of interest. Sometimes it is necessary to work a little harder in order to get a convenient parameterization. The key is to choose the identifiability constraints in order to obtain an interpretable set of coefficients. This can be done in statistical software by selecting different *contrasts*. We will not go further into this here, see the course textbooks or `?contrasts` for more information. The `relevel` function in R, as we have already learned, is also very useful!

6.7 ANOVA tables

In R the `anova` function produces tabular output related to the F-ratio comparison of alternative models. It is important to know exactly what the entries in these *ANOVA tables* mean, since they are a standard way of displaying results.

As seen in section 5.1.2, `anova` does different things depending on whether you call it with a single model argument, or several models. Consider the multiple model case first, using two of the `warpbreaks` models.

```
anova(wm1, wm)

## Analysis of Variance Table
##
## Model 1: sqrt(breaks) ~ wool + tension
## Model 2: sqrt(breaks) ~ wool * tension
##   Res.Df    RSS Df Sum of Sq    F  Pr(>F)
## 1     50 53.291
## 2     48 46.089  2    7.2014 3.75 0.03067 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Working across the columns of the ANOVA table...

- `Res.Df` gives the residual degrees of freedom for each model: i.e. the number of data (n) less the number of model parameters (p_0 and p_1, say, for the null and full models, respectively).
- `RSS` gives the residual sum of squares for each model (i.e. $\mathrm{RSS}_0 = \|\mathbf{y} - \mathbf{X}_0\hat{\boldsymbol{\beta}}_0\|^2$ and $\mathrm{RSS}_1 = \|\mathbf{y} - \mathbf{X}_1\hat{\boldsymbol{\beta}}_1\|^2$).
- `Df` gives the *difference* in degrees of freedom between each model and the model above it in the table (from 2nd model onwards). i.e. $p_1 - p_0$ in the current case.
- `Sum of Sq` gives the *difference* in residual sum of squares between each model and the model above it in the table (from 2nd model). In terms of section 4.3.2 this is $\mathrm{RSS}_0 - \mathrm{RSS}_1$.
- `F` is the F-ratio test statistic for comparing each model to the model above it in the table. That is $(\mathrm{RSS}_0 - \mathrm{RSS}_1)/\{(p_1 - p_0)\hat{\sigma}^2\}$. Note that $\hat{\sigma}^2$ is always calculated from the largest model in the table.
- `Pr(>F)` is the probability of getting an F-ratio at least as large as that in the `F` column if the model corresponding to the next row up in the table is correct. i.e. this is the p-value for testing the null hypothesis that the next model up is correct, against the larger alternative described by this row's model.

Only two models are compared in the example, but the definitions apply to any number of models from 2 upwards.

The second form of the ANOVA table occurs when only one model is supplied to `anova`. Here is an example.

```
anova(wm)

## Analysis of Variance Table
##
## Response: sqrt(breaks)
##              Df Sum Sq Mean Sq F value   Pr(>F)
## wool          1  2.902  2.9019  3.0222 0.088542 .
## tension       2 15.892  7.9458  8.2752 0.000817 ***
## wool:tension  2  7.201  3.6007  3.7500 0.030674 *
## Residuals    48 46.089  0.9602
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In this case the table should be read from the bottom up.

- The `Residuals` row gives the residual degrees of freedom, residual sum of squares and residual variance estimate, $\hat{\sigma}^2$, for the full model (`breaks~wool*tension`).
- The `wool:tension` row gives the *difference* in degrees of freedom and residual sum of squares between the full model and a model without the interaction (i.e. `breaks~wool+tension`). In some sense these numbers are the degrees of freedom and sum of squares associated with the dropped interaction term. The `Mean Sq` column gives the difference in sum of squares divided by the difference in degrees of freedom (an example of an $(\text{RSS}_0 - \text{RSS}_1)/(p_1 - p_0)$ term in section 4.3.2 notation). The `F value` is then the `Mean Sq` over $\hat{\sigma}^2$: the F-ratio for testing the null hypothesis that the model without the interaction is adequate, against the alternative that the full model is correct. The corresponding p-value is the last entry on the row.
- The `tension` row compares the model with wool and tension main effects to a model with just a wool effect. (i.e compares `breaks~wool+tension` to `breaks~wool`). The interpretation of the entries is as for the `wool:tension` row, but note that all the differences are taken between the two models being compared, not between the wool only model and the full model. $\hat{\sigma}^2$ in the F-ratio statistic still comes from the full model/ last row of the table, however.
- The `wool` row compares a model with wool only to a model with just an intercept (i.e `breaks~wool` to `breaks~1`). The interpretation of the entries is as for the previous row.

When interpreting such tables it is important to note two things:

- If you conclude that the interaction of two factors is significant and therefore can not be dropped from the model, then that means that the two factors concerned are significant, no matter what the p-values are for the rows concerning the factors separately. This is because the p-values for the main effects are only meaningful if their interaction is not significant (more on this shortly).
- For most models, the ANOVA table will differ depending on the order in which terms are removed from the model: this is because estimators for model terms are generally not independent. Notable exceptions are models involving factor variables fitted to *balanced* data. In these cases the terms estimators are generally independent, so that the order in which the the model is reduced is immaterial: the `warpbreaks` data are an example of this. Data are *balanced* w.r.t. a factor or interaction if there are the same number of observations per level of the factor or interaction.

6.7.1 Example of erroneous dropping of a main effect

To emphasize the importance of reading ANOVA table from the bottom up, and of not trying to drop main effects while leaving in interactions, here is what happens if one attempts to remove the `wool` main effect, while leaving in the `wool:tension` interaction...

```
wm2 <- lm(sqrt(breaks) ~ tension + tension:wool, data = warpbreaks)
summary(wm2)

##
## Call:
## lm(formula = sqrt(breaks) ~ tension + tension:wool, data = warpbreaks)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.69410 -0.70129  0.01772  0.66046  1.81902
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      6.5476     0.3266  20.046  < 2e-16 ***
## tensionM        -1.7216     0.4619  -3.727 0.000511 ***
## tensionH        -1.6912     0.4619  -3.661 0.000625 ***
## tensionL:woolB  -1.3094     0.4619  -2.835 0.006692 **
```

```
## tensionM:woolB   0.4727     0.4619   1.023 0.311278
## tensionH:woolB  -0.5542     0.4619  -1.200 0.236140
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9799 on 48 degrees of freedom
## Multiple R-squared:  0.3606,Adjusted R-squared:  0.294
## F-statistic: 5.415 on 5 and 48 DF,  p-value: 0.0004998

anova(wm2, wm)

## Analysis of Variance Table
##
## Model 1: sqrt(breaks) ~ tension + tension:wool
## Model 2: sqrt(breaks) ~ wool * tension
##   Res.Df    RSS Df   Sum of Sq F Pr(>F)
## 1     48 46.089
## 2     48 46.089  0 -3.5527e-14
```

The models have exactly the same RSS and residual degrees of freedom, so `anova` can't do anything sensible. Actually they are really the same model, so there is nothing to test. The following demonstrates the equivalence:

```
range(fitted(wm) - fitted(wm2))

## [1] -7.105427e-15  3.552714e-15
```

all that differs between them is the identifiability constraints, and hence the interpretation of the parameters. By removing the `wool` main effect all we have done is to remove the need for one of the constraints on the interaction, so an extra parameter of the interaction now does the work of the missing main effect parameter.

6.7.2 `drop1` again

Actually `drop1` is often a more sensible function to use than `anova` with a single argument. `drop1` will never drop main effects if their interaction is present, and is always comparing the omission of a single term, to the full model, rather than successively dropping more and more terms. Here it is applied to the `wm` model ...

```
drop1(wm, test = "F")

## Single term deletions
##
## Model:
## sqrt(breaks) ~ wool * tension
##              Df Sum of Sq    RSS    AIC F value  Pr(>F)
## <none>                    46.089 3.4461
## wool:tension  2    7.2014 53.291 7.2859    3.75 0.03067 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

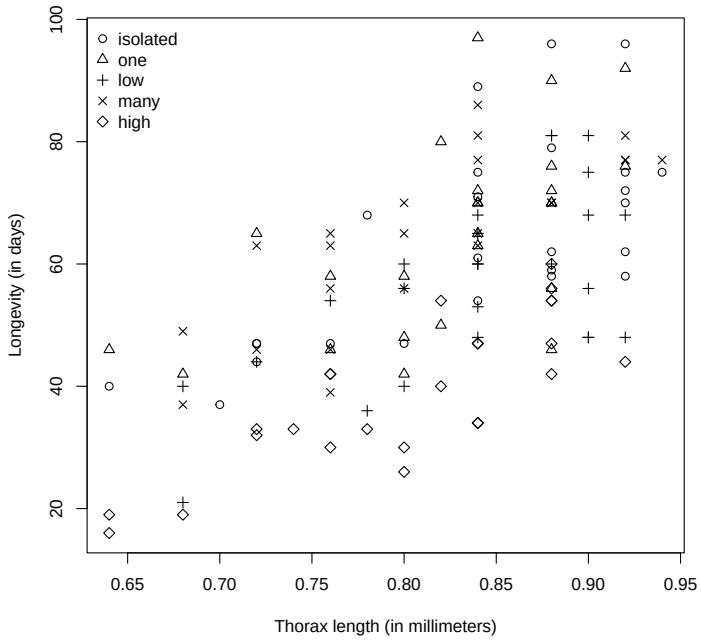
6.8 A model with both continuous and factor predictors

Of course, it is also possible to have both, possibly several, continuous and factor predictors. We will illustrate this with an example from Faraway (2014, p. 213) on a study on the sexual activity and life span of male fruitflies by Partridge and Farquhar (1981): 125 fruitflies were divided randomly into five groups of 25 each. The response was the longevity of the fruitflies in days. One group was kept solitary, while another was kept individually with a virgin female each day. Another group was given eight virgin females per day. As an additional control, the fourth and fifth groups were kept with one or eight pregnant females per day. Pregnant fruitflies will not mate. The thorax length of each male was measured as this was known to affect longevity. The five groups are labeled as

- *isolated*: fly kept with one pregnant fruitfly.
- *one*: fly kept with one pregnant fruitfly.
- *many*: fly kept with eight pregnant fruitflies.
- *low*: fly kept with one virgin fruitfly.
- *high*: fly kept with eight virgin fruitflies.

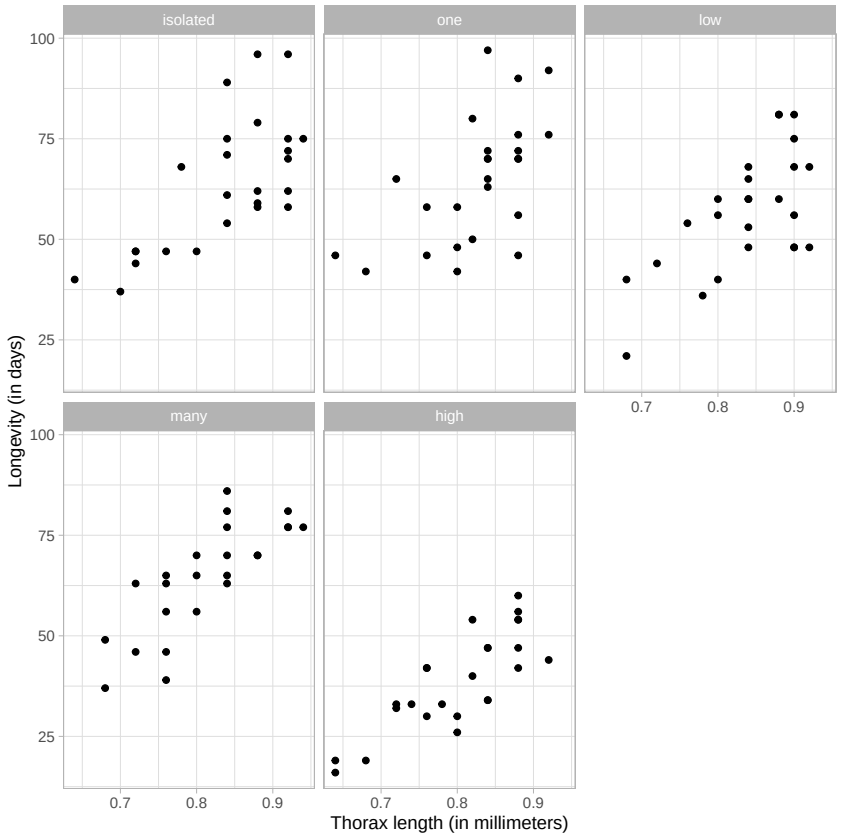
Note that the thorax length is the continuous predictor variable. The goal of the study was to see if sexual activity has a cost to male fruitflies, in terms of leongevity. We start with a plot of the data.

```
library(faraway)
levels(fruitfly$activity)

## [1] "isolated" "one"      "low"      "many"     "high"

plot(longevity ~ thorax, fruitfly, pch = unclass(activity),
     ylab = "Longevity (in days)", xlab = "Thorax length (in millimeters)")
legend("topleft", levels(fruitfly$activity), pch = 1:5, bty = "n")
```



Longevity for the high sexual activity group appears to be lower. With multiple levels, it can be hard to distinguish the groups. Sometimes it is better to plot each level separately.

```
ggplot(aes(x = thorax, y = longevity), data = fruitfly) +
  geom_point() +
  facet_wrap( ~ activity) +
  theme_light() +
  ylab("Longevity (in days)") +
  xlab("Thorax length (in millimeters)")
```



This plot makes clear that longevity for the high sexual activity group is lower.

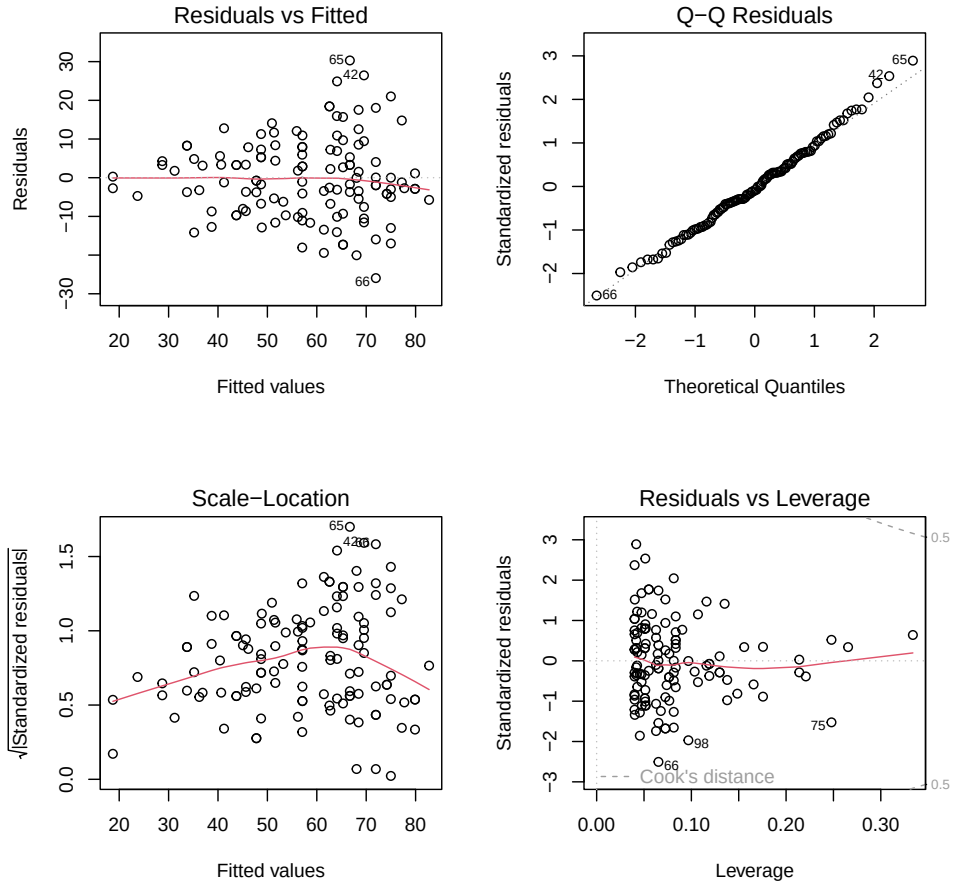
Two questions of interest are whether there is a difference in longevity between activity groups and whether the relationship between thorax and longevity is the same in each group. In order to be able to answer the questions of interest we fit a model containing the interaction between the thorax length and the activity group.

```
longm <- lm(longevity ~ thorax*activity, data = fruitfly)
head(model.matrix(longm))
```

```
##   (Intercept) thorax activityone activitylow activitymany activityhigh
## 1           1   0.68           0           0            1            0
## 2           1   0.68           0           0            1            0
## 3           1   0.72           0           0            1            0
## 4           1   0.72           0           0            1            0
## 5           1   0.76           0           0            1            0
## 6           1   0.76           0           0            1            0
##   thorax:activityone thorax:activitylow thorax:activitymany thorax:activityhigh
## 1                  0                  0                0.68                   0
## 2                  0                  0                0.68                   0
## 3                  0                  0                0.72                   0
## 4                  0                  0                0.72                   0
## 5                  0                  0                0.76                   0
## 6                  0                  0                0.76                   0
```

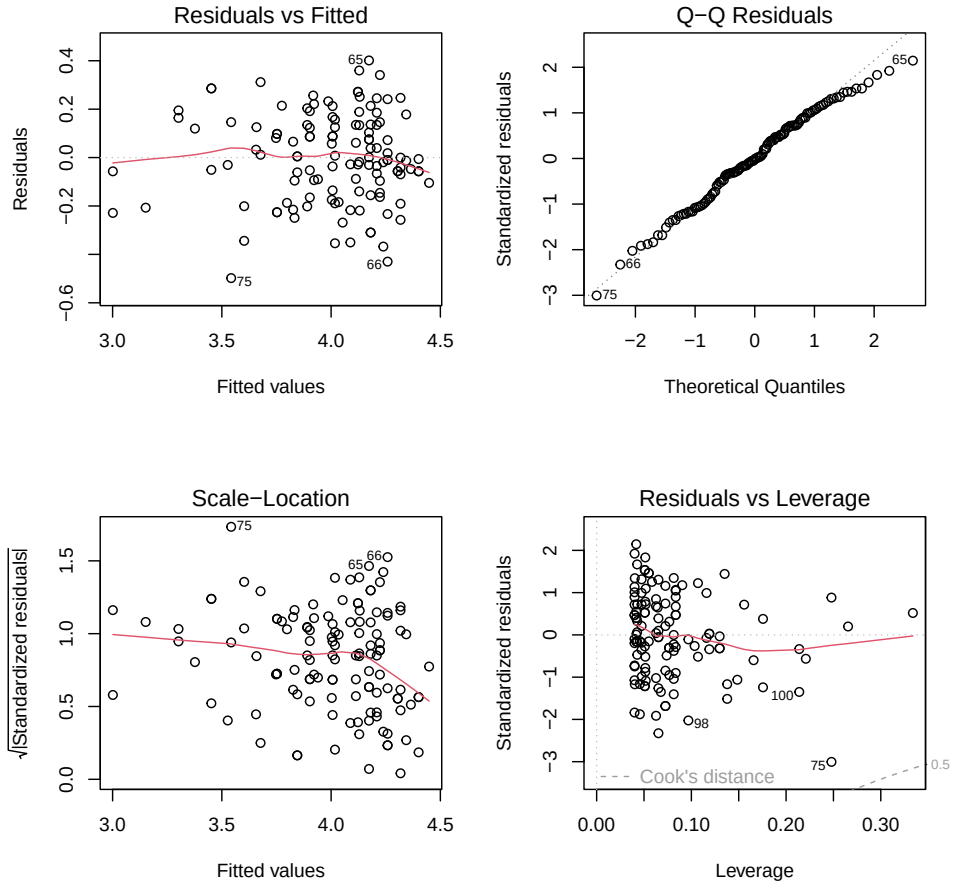
The reference level for the activity factor predictor is the isolated group Let us now check the residual plots.

```
par(mfrow = c(2, 2))
plot(longm)
```



The residuals versus fitted values plot show a clear funnel shape trend, indicating violation of the constant variance assumption. Upon inspecting both the square root and log transformations of the response data, it appeared that the log transformation worked better. Therefore, we will proceed with this transformation.

```
longm_log <- lm(log(longevity) ~ thorax*activity, data = fruitfly)
par(mfrow = c(2, 2))
plot(longm_log)
```



We can now proceed and check whether the interaction term is significant.

```
longm_log_1 <- lm(log(longevity) ~ thorax + activity, data = fruitfly)
anova(longm_log_1, longm_log)

## Analysis of Variance Table
##
## Model 1: log(longevity) ~ thorax + activity
## Model 2: log(longevity) ~ thorax * activity
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1    118 4.4012
## 2    114 4.1626  4   0.23857 1.6334 0.1706
```

The p-value is 0.1706 and so we do not reject the null hypothesis that the coefficients associated with interaction term are zero. That is, there is no evidence that the effect of thorax length on longevity differs between activity groups. As seen before, we could use the `drop1` function for the same purpose.

```
drop1(longm_log, test = "F")

## Single term deletions
##
## Model:
## log(longevity) ~ thorax * activity
##                 Df Sum of Sq    RSS     AIC F value Pr(>F)
## <none>                       4.1626 -400.87
## thorax:activity  4   0.23857 4.4012 -401.96  1.6334 0.1706
```

Can the model be further simplified?

```
drop1(longm_log_1, test = "F")

## Single term deletions
##
## Model:
## log(longevity) ~ thorax + activity
##          Df Sum of Sq    RSS     AIC F value    Pr(>F)
## <none>                4.4012 -401.96
## thorax    1    5.0757 9.4769 -308.86 136.086 < 2.2e-16 ***
## activity  4    4.1542 8.5554 -327.54  27.845 2.756e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Both AIC and hypothesis tests agree that the model containing the additive effects of thorax and activity variables is preferable to a model containing only one of these variables. Let us now inspect the summary output of the reduced model (containing only the additive effect of thorax length and activity).

```
summary(longm_log_1)

##
## Call:
## lm(formula = log(longevity) ~ thorax + activity, data = fruitfly)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.52641 -0.13629 -0.00823  0.13918  0.39273
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.84421    0.19882   9.276 1.04e-15 ***
## thorax        2.72146    0.23329  11.666  < 2e-16 ***
## activityone   0.05174    0.05468   0.946   0.3459
## activitylow  -0.12387    0.05463  -2.268   0.0252 *
## activitymany  0.08791    0.05546   1.585   0.1156
## activityhigh -0.41925    0.05527  -7.586 8.35e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1931 on 118 degrees of freedom
## Multiple R-squared:  0.7025,Adjusted R-squared:  0.6899
## F-statistic: 55.72 on 5 and 118 DF,  p-value: < 2.2e-16
```

Because of the log transformation, we can interpret the coefficients as having a multiplicative effect.

```
exp(coef(longm_log_1)[3:6])

##  activityone  activitylow activitymany activityhigh
##    1.0531064    0.8834971    1.0918894    0.6575384

exp(confint(longm_log_1)[3:6, ])

##                  2.5 %    97.5 %
## activityone  0.9450347 1.1735370
## activitylow  0.7929135 0.9844290
## activitymany 0.9783179 1.2186452
## activityhigh 0.5893739 0.7335865
```

Compared to the reference level (isolated group), we see that the high sexual activity group has 0.66 times the life span (i.e., 34% less). And note that 1 ($= e^0$) is not included in the corresponding confidence interval.

7 How to approach an analysis

By now we have seen almost all the theory we need for practical analysis, and how to put much of it into practice. It's time to stand back, take stock, and think about some general principles for approaching any modelling task. It is not possible, or desirable, to produce a recipe that all analyses should follow, but here are some guidelines.

1. Always start by identifying the questions that you are trying to answer by analysing the data.
2. Always look at the data before fitting any model. Plot the data to get a feel for how variables are related. Check for obvious errors. If you can think of simple methods (e.g. plots) that will give you informal answers to your questions, use them before starting the formal model based analysis.
3. Now think about how you can use linear models to answer the questions of interest.
    - Sometimes at least the main features of the models and analysis are obvious from the question. For example, analysing clinical trial data to test whether a new drug is better than the standard treatment will obviously involve comparing models with and without the treatment factor, by hypothesis testing, and then estimating the size of any effect differences.
    - Other times the question is much less prescriptive: as in the tyre wear example, answering the question of interest may involve finding a linear model that adequately captures the relationship between the response and predictors, but the question and wider context do not specify much about what model to use. In such cases you usually have to think about broad classes of models that might be sensible, and then use statistical model selection approaches to choose between them.
    - When formulating a model, ensure that you are clear what is the response, and what are the predictors.
    - Sometimes it is clear in advance that not all potential predictors should be included in the model. But be careful. Don't include variables that you know are irrelevant, but don't pre-exclude variables unless you are pretty certain about this.
    - Try to avoid excluding predictors on the basis of statistical analysis of the predictors alone (i.e. without consideration of the response). Of course this advice does not extend to the case when two predictors actually measure the same thing: then it is sensible to either combine them, or decide which one to drop.
4. Once you start fitting the models that are part of your analysis, make sure that you check that the modelling assumptions are met. Hypothesis tests, confidence intervals and so on don't work if they are not.
5. Always make sure that you interpret the results of your modelling in terms of the original question, and think carefully about any limitations that apply to the answer.

Finally, some principles for writing up a statistical analysis

1. Always clearly explain the context, and the questions being addressed.
2. Make sure that your analysis is repeatable by any statistician with access to the data, using the software of their choice. This means presenting models and results in universal mathematical language, not as R code and output.
3. Always explain the 'why' of your analysis as well as the 'what'.
4. Relate your conclusions back to the questions, and be careful to discuss the limitations of the approach taken.
5. Aim to be concise and useful, and at all costs avoid the sort of legalistic back covering approach in which every analysis report lists every possible thing that could ever go wrong with a linear model. Such an approach tells the reader nothing about what might be a real issue with the analysis actually being reported.

8 Causality, confounding and randomization (non-examinable)

The material in this section is non-examinable; however, please do not skip this section. It is only a few pages, and it is both important and instructive.

There are several different purposes for which we may wish to use a (generalized) linear model.

1. For prediction. For example we build a model of patients' probability of heart disease based on lifestyle factors such as diet and exercise, estimate the model from a representative group of subjects whose heart disease status is known, and use it to predict the status of new subjects.

2. To establish association between variables. For example we suspect that being educated in a single sex school may be negatively associated with length of marriage, and use statistical modelling to assess whether this putative association is real, or could just be due to chance variability in the data.

3. To establish causation. Often, especially in science, we want to know whether something is an actual cause of something else. Famous examples are: does smoking cause cancer? and does the MMR (measles, mumps, rubella) vaccine cause autism?

The methods we have covered so far are quite sufficient for addressing 1 and 2, but the *causal inference* required in 3 is more difficult, and the sort of methods covered so far (and that will be covered in the remaining sections) are not sufficient to establish causality *on their own*.

To understand the issues, it is important to really have grasped the difference between correlation (association) and causality. An oft quoted example is the correlation between the population of storks and the birth rate, in Europe. The correlation is real, but there is obviously no causation. Rather, industrialization raised health care, living and educational standards in Europe leading to reduced birth rates, but that same industrialization led to a reduction in suitable habitat for storks. That is, the association is caused by other factors which may be causative to both variables of interest. To put it simply, for an association to reflect causality, it must not have an *alternative explanation*. We may think we have found a causal relationship, but we may merely not have thought of a particular reason that can explain the association.

Confusing correlation with causation is a famously good way of selling newspapers. For example, some medical studies have found associations between coffee drinking and various responses, such as the likelihood of a heart attack or the mean blood pressure level. But after taking into account other variables associated with the extent of coffee drinking, such as extent of smoking, country residence, occupation, and levels of stress, such associations have disappeared or weakened considerably. Again the key is (are) a hidden variable(s) related to the variables of direct interest.

To appreciate the effect of correlated predictors on interpretation, and on attempts at causal inference, it is worth looking at some simulations, where we are in complete control of the correlation structure between variables. The following code simulates two highly correlated (correlation of 0.95) predictor variables, x and h:

```
#supplies rmvn to simulate random numbers from a multivariate normal dist.
require(mgcv)
V <- matrix(c(1, 0.95, 0.95, 1), 2, 2) #covariance matrix
n <- 1000 #number of data points
set.seed(7) #just for reproducibility
#predictors simulated from a multivariate (bivariate) normal dist.
X <- rmvn(n, rep(0, 2), V)
x <- X[ , 1]; h <- X[ , 2]
```

Now consider the case when $y_i = x_i + \epsilon_i$, but we fit the model $y_i = \beta_0 + \beta_1 x_i + \beta_2 h_i + \epsilon_i$.

```
#simulate response variable as specified above
y <- x + rnorm(n)
summary(lm(y ~ x + h))$coefficients

##                Estimate Std. Error    t value     Pr(>|t|)
## (Intercept) -0.003268241 0.03116981 -0.1048528 9.165137e-01
## x            0.812567847 0.09892433  8.2140345 6.587810e-16
## h            0.209603373 0.09947463  2.1071038 3.535786e-02
```

Because of the high correlation between x and h it is difficult to distinguish their effects, with the result that when both are included in the model the 'explanation' of the response tends to get shared out between them, leading to rather variable estimates of the real effect of x. In contrast when the spurious h term is omitted from the model, the estimate of β_1 is much more accurate. Actually, for most replicates of the data simulation (by specifying a different seed for the random number generator), β_2 would not have appeared to be significantly different from zero, so we would have been likely to drop the term, and arrive at the correct model using just the inferential tools already developed.

However, it is worth also looking at what would have happened if we had only observe h but not x (remember that y only depends on x and not on h):

```
summary(lm(y ~ h))$coefficients

##                 Estimate Std. Error    t value      Pr(>|t|)
## (Intercept) -0.007192532 0.03218731 -0.2234586  8.232243e-01
## h            0.985160671 0.03233624 30.4661526 1.153475e-144

cor(y, h)

## [1] 0.6941742
```

The high correlation between h and x means that we find a strongly significant association between h and y, despite the fact that h played no part at all in the simulation of y! In this case x is an example of a hidden *confounding variable*: it is correlated with both our predictor, h, and our response y, and therefore messes up our inference about the true causal influence of h on y (which is zero in this case).

Here is a second example of confounding. Now both h and x have a causal effect on y, and if we only observe x, we will overestimate its true effect (overestimate, because the correlation is positive).

```
y <- x + h + rnorm(n)
summary(lm(y ~ x ))$coefficients

##               Estimate Std. Error    t value  Pr(>|t|)
## (Intercept) 0.01714281 0.03248967  0.5276387 0.5978674
## x           1.92019944 0.03246061 59.1547598 0.0000000

summary(lm(y ~ x + h))$coefficients

##               Estimate Std. Error    t value     Pr(>|t|)
## (Intercept) 0.01257442 0.03144589  0.3998749 6.893343e-01
## x           1.13515597 0.09980053 11.3742474 2.850763e-28
## h           0.83168316 0.10035571  8.2873525 3.709458e-16
```

That is, when h is hidden from us, our estimate of β_1 is almost twice as large as it should be in this case.

Note that this tendency of the model fitting to try and compensate for missing confounding variables, is a profound difficulty for causal inference, but a blessing for prediction. The fact that the model has adjusted for the missing confounders in this way actually improves prediction.

The gold standard for establishing causality is to use data from a properly designed experiment, in which we use the process of *randomization* to break any possible association between unmeasured confounders and the predictor whose effect we want to measure. The idea is to turn the systematic variation in the response caused by confounders into something that we can model as independent random variability between experimental units.

An example conveys the idea. Suppose that we want to examine the relationship between exercise and fat mass in people aged 40 to 50. There are a huge number of factors contributing to people's fat mass, and if we simply look at a sample of people who already exercise by different amounts it will be very difficult to isolate the effect of exercise separate from all the other confounders (diet, working hours, profession, whether they have children, drinking habits, smoking, wealth etc.). Suppose instead that we set up a study in which a large enough number of participants enrol and are then assigned to one of several 'treatments', consisting of exercise programs of differing intensity (or a control of none). Now if we assign subjects to the different treatments randomly then

there can be no possible association between all the other variables contributing to fat mass, and the one that we are interested in: amount of exercise. Indeed all the variability in fat mass attributable to the confounders can now be treated as random variability within each treatment. Any effect of exercise that we then find is causal: the thing we controlled having an effect on the variable we are interested in. Of course this approach does not stop us from including measured covariates in the analysis model, in order to reduce the amount of variability being modelled as random, and thereby increase precision, but it does mean that we no longer have to worry about the effect of hidden confounders.

So properly designed controlled experiments with random allocation of experimental units to treatments are the best way of establishing causal effects. However there are many cases in which experimental manipulation is impractical, unethical and/or illegal. For example, if we are interested in establishing the causative effect of alcohol consumption on heart disease, it is simply unethical to conduct a study in which we randomly allocate subjects to a heavy drinking treatment (the same goes for any treatment where we have prior cause to suspect it may be harmful). Economics is another field beset with such problems: for example it is impractical to conduct a controlled experiment to establish the causative factors controlling a company's share price, and anything that came close would likely count as illegal market manipulation. This sort of problem is so ubiquitous in economics that it is from the field of *econometrics* that much of the work on causal inference from observational data comes. Depending on the field (e.g.,medicine versus economic or social sciences) different techniques are more prevalent than others, but popular approaches are instrumental variables, differences in differences, regression discontinuity designs, G-computation, propensity scores, etc. These are, however, out of the scope of this course. Causal inference is currently a topic of active research. Indeed, the 2021 Economic Nobel prize was awarded to Guido Imbens who does foundational research on causal inference (https://www.nobelprize.org/prizes/economic-sciences/2021/imbens/facts/). The following short book is a very nice introduction to the topic

https://mitpress.mit.edu/9780262545198/causal-inference/

Finally, the following website contains several fun examples of spurious correlations:

https://www.tylervigen.com/spurious-correlations

9 Maximum likelihood estimation: a brief review

9.1 Why maximum likelihood estimation now?

It is time to move on and consider some more general models inspired by the linear model. Indeed, in many situations, a Gaussian distribution is inappropriate as a model for a response variable of interest, although a linear model structure might be appropriate for the expected value of the response, leading to a *generalized linear model* (GLM). For example a discrete response variable might follow a Poisson or binomial distribution, or a positive continuous response variable might be better described by a gamma distribution. Our existing theory is then insufficient: for a start all of these distributions break the linear model constant variance assumption. However, without actually straying very far from the linear model, we therefore have an obvious motivation to develop inference methods that apply beyond the linear model. The general theory of *maximum likelihood estimation* provides the key to this.

9.2 Maximum likelihood estimation

The key idea of *maximum likelihood estimation* is simply this:

> Parameter values that make the observed data appear relatively probable are more *likely* to be correct than parameter values that make the observed data appear relatively improbable.

For example, we would much prefer an estimate of parameter vector, $\boldsymbol{\theta}$, that assigned a probability density of 0.1 to our observed $\mathbf{y}$, according to the model, to an estimate for which the density was 0.00001.

So the idea is to judge the *likelihood* of parameter values using $f_\theta(\mathbf{y})$, the model probability density or mass function, according to the given value of $\boldsymbol{\theta}$, evaluated at the observed data. Because $\mathbf{y}$ is now fixed and we are considering the likelihood as a function of $\boldsymbol{\theta}$, it is usual to write the likelihood as $L(\boldsymbol{\theta}) \equiv f_\theta(\mathbf{y})$. In fact, for theoretical and practical purposes it is usual to work with the log likelihood $l(\boldsymbol{\theta}) = \log L(\boldsymbol{\theta})$. The *maximum likelihood estimate* (MLE) of $\boldsymbol{\theta}$ is then

$$\hat{\boldsymbol{\theta}} = \underset{\theta}{\operatorname{argmax}}\, l(\boldsymbol{\theta}).$$

In general, there will be no closed form solution for $\hat{\boldsymbol{\theta}}$ and numerical methods will be needed to actually find $\hat{\boldsymbol{\theta}}$, but reliable and general methods are available.

There is more to maximum likelihood estimation than just its intuitive appeal. Under some regularity conditions and in the large sample limit as $n \to \infty$,

$$\hat{\boldsymbol{\theta}} \sim N(\boldsymbol{\theta}, \boldsymbol{\mathcal{I}}^{-1}), \tag{16}$$

where $\boldsymbol{\mathcal{I}} = -\mathbb{E}(\partial^2 l/\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^\mathsf{T})$ — the expected second derivative matrix of the negative log likelihood (also known as the *information matrix*). This turns out to be the best that can be achieved for an unbiased estimator (although MLEs are only unbiased in the large sample limit). These results are closely related to the fact that $\partial l/\partial\boldsymbol{\theta}$ has expected value $\mathbf{0}$ and covariance matrix $\boldsymbol{\mathcal{I}}$. In fact a similar result applies in terms of the observed (rather than expected) second derivative matrix. If $\hat{\boldsymbol{\mathcal{I}}} = -\partial^2 l/\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^\mathsf{T}$ (evaluated at $\hat{\boldsymbol{\theta}}$) then in the $n \to \infty$ limit,

$$\hat{\boldsymbol{\theta}} \sim N(\boldsymbol{\theta}, \hat{\boldsymbol{\mathcal{I}}}^{-1}).$$

Another large sample result is also useful. Suppose that we want to compare two *nested* models, meaning that one model can be written as the other model subject to r restrictions on its parameters, $\boldsymbol{\theta}$, that can be written as $\mathbf{R}(\boldsymbol{\theta}) = \mathbf{0}$. Let $\hat{\boldsymbol{\theta}}_0$ denote the MLEs under the restriction. Then we can test $H_0 : \mathbf{R}(\boldsymbol{\theta}) = \mathbf{0}$ (i.e. test the null hypothesis that the restricted model is correct), by using the large sample result that

$$2\{l(\hat{\boldsymbol{\theta}}) - l(\hat{\boldsymbol{\theta}}_0)\} \sim \chi^2_r \tag{17}$$

if the null hypothesis is correct. If the null hypothesis is not correct then the log likelihood ratio statistic on the left hand side will tend to be too large for consistency with a χ^2_r distribution. As usually, to actually judge compatibility of data and null hypothesis we compute a p-value — the probability of a χ^2_r random variable being at least as large as the observed test statistic. As usual a small p-value is evidence against the null hypothesis. Tests conducted

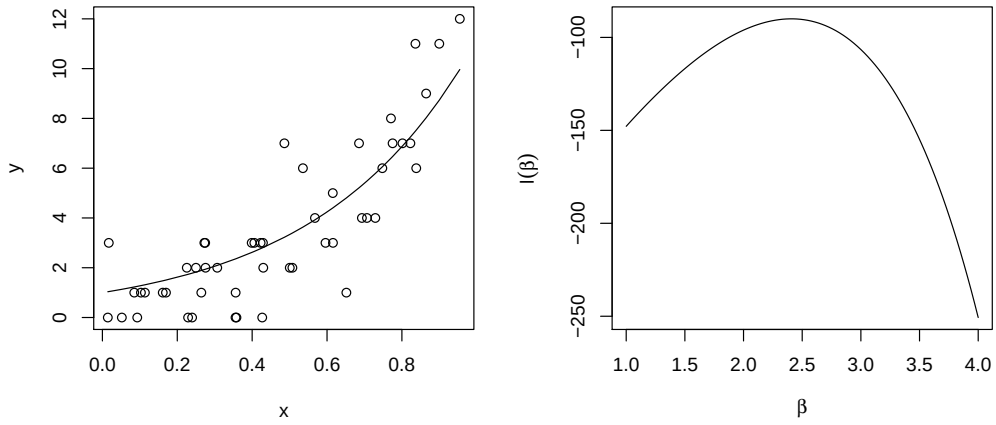


Figure 6: Left: 50 Poisson data, y, dependent on predictor data, x, and described by the model in section 9.2.1. The black curve is the maximum likelihood estimate of the Poisson mean as a function of x. Right: the log likelihood function for the model parameter β, computed using the data in the left panel and as described in the text.

using this result are known as *generalized likelihood ratio tests* (GLRT). Likelihood ratio tests are similar to the F-tests we learned in section 4.3.2, in the sense they compare the full model with a restricted model where some predictors (or even just a factor predictor) are omitted.

Deriving these results is beyond the scope of this course but, for example, Wood (2015, chapter 4) *Core Statistics* provides compact derivations. Further, most of these results, if not all, were also proved in *Statistical Methodology* (MATH10095).

As it was the case for F-tests, hypothesis tests like the GLRT offer one way to choose between models, essentially favouring the simpler model unless the data offer sufficient evidence that this is not tenable and a more complicated model is needed. As we already learned in section 5, an alternative is to try to estimate how close the model is too the 'true' model and choose the model that is closest. This approach leads to the selecting the model that has the smallest value of the Akaike Information Criterion among the candidate models. Recall that for a given model, the AIC is given by $-2l(\hat{\boldsymbol{\theta}}) + 2p$, with p being the number of model parameters - the dimension of $\boldsymbol{\theta}$.

9.2.1 A simple example

Consider the one parameter Poisson model:

$$y_i \overset{\text{ind}}{\sim} \text{Poisson}\{\exp(\beta x_i)\},$$

and suppose that we have n pairs of (x_i, y_i) data. Generally the probability function for a Poisson random variable with mean λ_i is $f(y_i) = \lambda_i^{y_i} \exp(-\lambda_i)/y_i!$. So for our model, for which $\lambda_i = \exp(\beta x_i)$, we have

$$f(y_i) = \exp(\beta x_i)^{y_i} \exp\{-\exp(\beta x_i)\}/y_i!.$$

To compute the likelihood we need to compute the joint probability of all the data. Because our model says that the data are independent, this joint probability is just the product of the individual probabilities for each y_i, i.e. the likelihood function is just

$$L(\beta) = \prod_{i=1}^{n} \exp(\beta x_i)^{y_i} \exp\{-\exp(\beta x_i)\}/y_i!,$$

and repeatedly using the facts that $\log(a^b) = b\log(a)$ and $\log(ab) = \log(a)+\log(b)$, the log likelihood is therefore

$$l(\beta) = \sum_{i=1}^{n} \{y_i \beta x_i - \exp(\beta x_i) - \log(y_i!)\}.$$

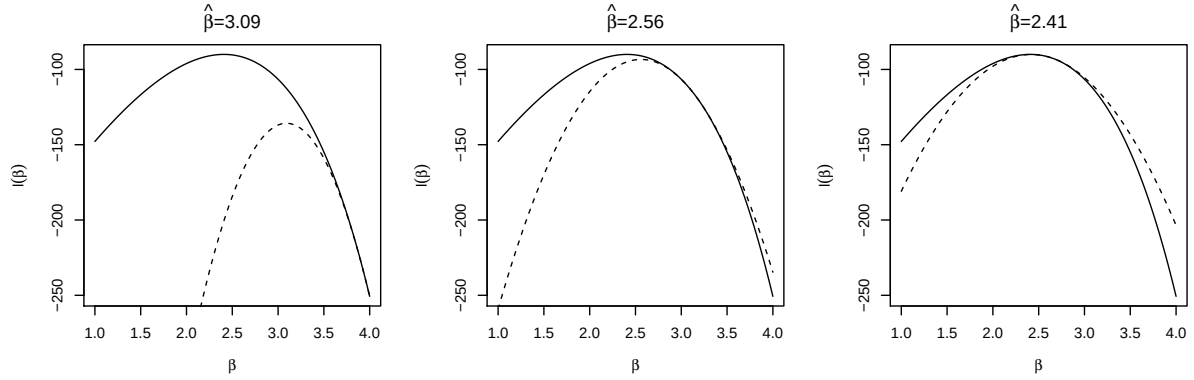


Figure 7: Three steps of Newton's method applied to the log likelihood for the Poisson example of section 9.2.1, starting on the left from $\beta = 4$. The dashed curve shows the approximating quadratic matching the value and first two derivatives of the log likelihood at the previous best estimate of β. The panel headings give the value, $\hat{\beta}$ maximising the approximating quadratic at each step. In this case no step length control is needed and the second derivative is always negative.

Figure 6 shows some (x, y) data (left) for which this model is appropriate, and plots the log-likelihood function for a range of β values (right). $\hat{\beta} = 2.41$ is the MLE, and the continuous curve overlaid on the left panel is the Poisson mean that this implies (i.e., $\exp(-\hat{\beta}x_i)$).

The key idea then, is that the log likelihood function provides us with a generic way of measuring how well a model fits data, and which parameter values do the best job at explaining the data. The simple Poisson example only considered a single parameter, but in principle we can compute the log likelihood for a model with as many parameters as we like, although we do need to be sure that we have sufficient data if we are intending to estimate large numbers of parameters.

9.2.2 Practical likelihood maximization

As this simple Poisson regression example already made clear, it is rarely possible to find explicit expressions for the maximum likelihood estimates, and instead we must maximize the log likelihood numerically. Newton's method is the gold standard for doing this, and works my optimizing successive quadratic approximations to the log likelihood. We start with an initial parameter value guess, and try to obtain a quadratic approximation to the log-likelihood that behaves like the log-likelihood itself in the vicinity of that guess. To do this we can use a Taylor approximation to the log-likelihood, discarding the terms above second order. That is we evaluate the log likelihood and its first two derivatives, or the first derivative vector and second derivative matrix in usual case of a parameter vector, at the parameter value (or vector), and then find the unique quadratic function which also has this value and derivatives at the parameter value. Hopefully the approximation has a maximum close to the maximum of the log-likelihood itself, and it is easy to write down an explicit form for the maximizer of the approximating quadratic, using this as our updated parameter guess. Iterating this procedure we eventually reach the maximum of the log likelihood itself.

The mathematical details about the Newton-Raphson method, for the most general case of a parameter vector $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^\mathsf{T}$, follow. Let

$$\mathbf{U}(\boldsymbol{\beta}) = \left(\frac{\partial l(\boldsymbol{\beta})}{\partial \beta_1}, \cdots, \frac{\partial l(\boldsymbol{\beta})}{\partial \beta_p} \right)^\mathsf{T}$$

be the score function. Let $\mathbf{H}(\boldsymbol{\beta})$ denote the matrix $\partial \mathbf{U}(\boldsymbol{\beta})/\partial \boldsymbol{\beta}$ having entries

$$h_{jk} = \frac{\partial^2 l(\boldsymbol{\beta})}{\partial \beta_j \partial \beta_k}, \quad j = 1, \ldots, p, \quad , k = 1, \ldots, p,$$

called a Hessian matrix. Step t in the iterative process ($t = 0, 1, 2, \ldots$) approximates $l(\boldsymbol{\beta})$ near $\boldsymbol{\beta}^{(t)}$ by a second-order Taylor series expansion,

$$l(\boldsymbol{\beta}) \approx l(\boldsymbol{\beta}^{(t)}) + \mathbf{U}(\boldsymbol{\beta}^{(t)})^{\mathsf{T}}(\boldsymbol{\beta} - \boldsymbol{\beta}^{(t)}) + \frac{1}{2}(\boldsymbol{\beta} - \boldsymbol{\beta}^{(t)})^{\mathsf{T}}\mathbf{H}(\boldsymbol{\beta}^{(t)})(\boldsymbol{\beta} - \boldsymbol{\beta}^{(t)}).$$

Applying rules for differentiating of vector functions and solving

$$\frac{\partial l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} \approx \mathbf{U}(\boldsymbol{\beta}^{(t)}) + \mathbf{H}(\boldsymbol{\beta}^{(t)})(\boldsymbol{\beta} - \boldsymbol{\beta}^{(t)}) = \mathbf{0}$$

for $\boldsymbol{\beta}$ yields

$$\boldsymbol{\beta}^{(t+1)} = \boldsymbol{\beta}^{(t)} - \mathbf{H}(\boldsymbol{\beta}^{(t)})^{-1}\mathbf{U}(\boldsymbol{\beta}^{(t)}).$$

Iterations proceed until changes in $l(\boldsymbol{\beta}^{(t)})$ between successive iterations are sufficiently tiny. There are, however, two modifications required to guarantee that Newton's method converges to the MLE. Firstly we need to ensure that the approximating quadratic is one that actually has a maximum (as opposed to a minimum, or even, in more than one dimension, a saddlepoint). In one dimension this is a matter of checking that the second derivative is negative and replacing it with a negative number if it isn't. In more dimensions there is also an equivalent fix (check that the negative of the second derivative matrix is positive definite, and if necessary modify it to force this condition to hold). The second modification is to check that the proposed change in parameter values actually increases the log likelihood itself. If it doesn't we just move the parameter back towards the previous parameter guess, until the log likelihood is increased at the new values. With these modifications Newton's method will find a maximum of the log likelihood, provided it has one.

Figure 7 illustrates Newton's method for the case of the single parameter Poisson regression model. In this case the maximum is reached in 3 steps (to graphical accuracy). Notice how the final quadratic matches the log likelihood in the vicinity of the MLE.

9.2.3 Illustrating $\hat{\boldsymbol{\theta}}$ uncertainty

Figure 8 shows log likelihood curves for 100 replicates of the data shown in the left panel of figure 6, when the true β is 2.5, along with the corresponding MLEs, $\hat{\beta}$, and the corresponding non-parametric (kernel density) estimate of the distribution of $\hat{\beta}$. Notice how the likelihood function and $\hat{\beta}$ vary from replicate to replicate, but how the distribution of $\hat{\beta}$ is approximately normal and centred on the true β value of 2.5. This is exactly what Equation (16) implies will happen.

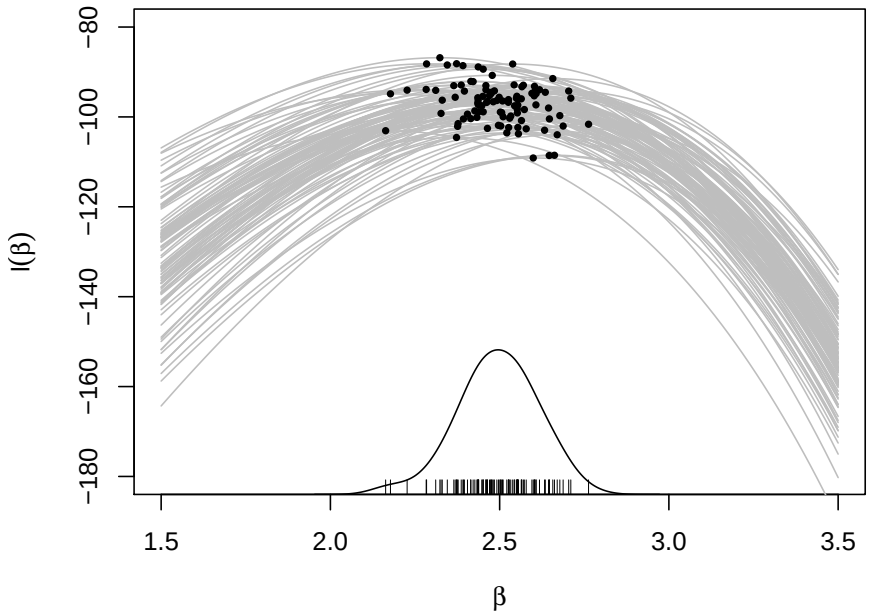


Figure 8: Log likelihood functions (grey) for 100 replicates of the section 9.2.1 Poisson dataset, when $\beta = 2.5$. The black dots show the maximum of each curve. The corresponding MLEs, $\hat{\beta}$, are shown as vertical bars on the x-axis (a rug plot). The continuous black curve is a (rescaled and vertically shifted) kernel density estimate of the distribution of $\hat{\beta}$ based on the shown 100 estimates. Notice how the distribution of $\hat{\beta}$ is approximately normal, with mean at the true value, $\beta = 2.5$, although there is zero probability that $\hat{\beta} = 2.5$ exactly.

10 Introducing generalized linear models

As we have seen extensively in the previous sections, a linear model is a statistical model that can be written as

$$y_i = \mathbf{X}_i\boldsymbol{\beta} + \epsilon_i, \quad \epsilon_i \overset{\text{i.i.d}}{\sim} N(0, \sigma^2).$$

Recall that an exactly equivalent way of writing this linear model is

$$\mathbb{E}(y_i) \equiv \mu_i = \mathbf{X}_i\boldsymbol{\beta}, \quad y_i \overset{\text{indep.}}{\sim} N(\mu_i, \sigma^2).$$

Generalized linear models extend linear models by allowing some non-linearity in the model structure and much more flexibility in the specification of the distribution of the response variable y_i. A GLM has the basic structure:

$$g(\mu_i) = \mathbf{X}_i\boldsymbol{\beta}, \quad \mu_i \equiv \mathbb{E}(y_i),$$

where g is a smooth monotonic (i.e., differentiable and invertible) *link function*, $\mathbf{X}_i$ is the ith row of the design matrix and $\boldsymbol{\beta}$ is a vector of unknown parameters. In addition, a GLM usually makes the distributional assumption that the y_i are independent and

$$y_i \sim \text{ some exponential family distribution.}$$

The *exponential family* of distributions includes many distributions that are useful for practical modelling, such as the Poisson, binomial, gamma, and normal distributions. A convenient property of distributions in the exponential family, with the exception of the normal distribution, is that the variance of y_i is a function of its mean μ_i. A GLM therefore consists of three components:

- The *systematic component*: $\mathbf{X}_i\boldsymbol{\beta}$.
- The *random component*: the specified distribution for y.
- The *link function* g.

Because generalized linear models are specified in terms of the *linear predictor* $\eta \equiv \mathbf{X}\boldsymbol{\beta}$, many of the general ideas and concepts of linear modelling carry over with a little modification, to generalized linear modelling. Basic model formulation is much the same as for linear models, except that a link function and a distribution for the response variable much be chosen. Of course, if the identity function is chosen as the link function, along with the normal distribution for the response variable y, then the normal linear model we have covered in sections 1–6 is recovered as a special case.

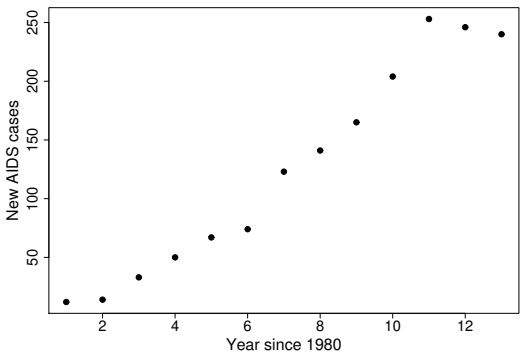
The generalization comes at some cost: model fitting now has to be done iteratively, and distributional results, used for inference, are now approximate and justified by large sample limiting results, rather than being exact.

A brief historical note: generalized linear modelling as a methodology was developed in 1972 by John Nelder and Robert Wedderburn while working at the Rothamsted Experimental Station in the UK.

Before going further into the methodological issues, let's consider some simple examples.

10.1 Simple single covariate examples of GLMs

Example 1: AIDS in Belgium.



The above figure shows new AIDS cases each year in Belgium, at the start of the epidemic. In the early stages of an epidemic an exponential increase model is often appropriate. Hence, if μ_i is the expected number of new AIDS cases per year and if t_i denotes the number of years since 1980, then a suitable model might be

$$\mathbb{E}(y_i) \equiv \mu_i = \gamma e^{\alpha t_i},$$

where γ and α are unknown parameters and y_i is the number of new AIDS cases per year. Such a model can be turned into GLM form by using the log link so that

$$\log(\mu_i) = \log(\gamma) + \alpha t_i = \beta_1 + \beta_2 t_i, \quad \beta_1 = \log(\gamma), \quad \beta_2 = \alpha.$$

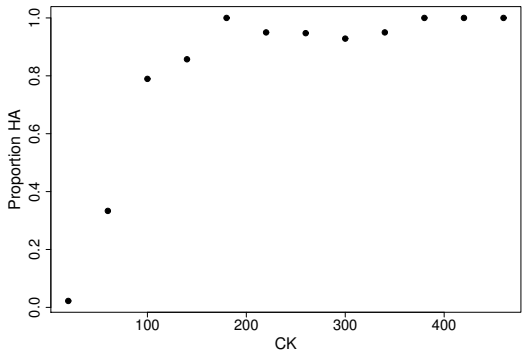
Note that the right hand side of the model is now linear in the parameters. The response variable as just mentioned is the number of new AIDS cases per year and, since this is a count, the Poisson distribution is probably a reasonable distribution to try. So the GLM for this situation uses a Poisson response distribution, log link, and linear predictor $\beta_1 + \beta_2 t_i$ (and thus the ith row of the design matrix is $\mathbf{X}_i = [1, t_i]$).

Example 2: Heart Attacks and Creatine Kinase

The following data are from a study examining the efficacy of blood creatine kinase (CK) levels as a diagnostic when patients present with symptoms that may indicate a heart attack.

CK level	20	60	100	140	180	220	260	300	340	380	420	460
Heart Attack	2	13	30	30	21	19	18	13	19	15	7	8
Not Heart Attack	88	26	8	5	0	1	1	1	1	0	0	0

Here is a plot of the proportion of patients who subsequently turned out to have had a heart attack, against their blood CK levels on admission to hospital.



Let p_i be the probability of heart attack at CK level x_i. A particularly convenient model for describing this probability is

$$p_i = \frac{e^{\beta_1 + \beta_2 x_i}}{1 + e^{\beta_1 + \beta_2 x_i}}.$$

This curve is sigmoid in shape, and bounded between 0 and 1. If y_i is the number of heart attack victims observed out of N_i patients with CK level x_i, we have that $y_i \sim \text{Binomial}(N_i, p_i)$ and therefore,

$$\mathbb{E}(y_i) \equiv \mu_i = N_i p_i = N_i \frac{e^{\beta_1 + \beta_2 x_i}}{1 + e^{\beta_1 + \beta_2 x_i}}.$$

This model is somewhat non-linear in its parameters, but if we apply the *logit* link function

$$g(\mu_i) = \log\left(\frac{\mu_i}{N_i - \mu_i}\right),$$

to both sides we obtain that

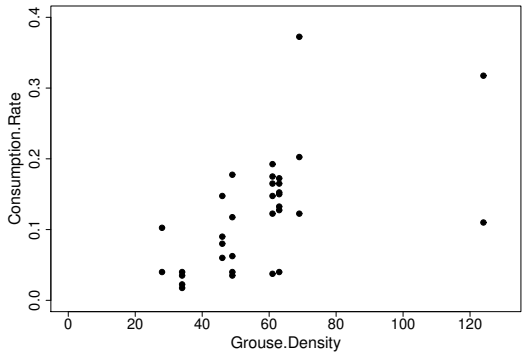
$$\log\left(\frac{\mu_i}{N_i-\mu_i}\right)=\beta_1+\beta_2 x_i, \tag{18}$$

which is clearly a GLM. Note that another way of writing (18) is

$$\log\left(\frac{\mu_i}{N_i-\mu_i}\right)=\log\left(\frac{N_i p_i}{N_i-N_i p_i}\right)=\log\left(\frac{p_i}{1-p_i}\right)=\beta_1+\beta_2 x_i.$$

This model is known as *logistic regression*.

Example 3: Hen harriers and Grouse



The above plot shows the daily consumption rate of Grouse by Hen Harriers (a type of bird of prey) plotted against the density of Grouse on various Grouse moors. If c_i denotes consumption rate and d_i is grouse density, then ecological theory suggests that the expected consumption rate should be related to grouse density by the following model

$$\mathbb{E}(c_i)\equiv\mu_i=\frac{\alpha d_i^3}{\delta+d_i^3},$$

where α and δ are parameters to be estimated. Note that as we shall see later when analysing this dataset, the cubic power can be replaced by other powers (AIC will help in choosing the right power for this dataset). Obviously this model in nonlinear in its parameters but by using the 'inverse' link function, the right hand side of the previous equation can be made linear in the parameters

$$\frac{1}{\mu_i}=\frac{1}{\alpha}+\frac{\delta}{\alpha}\frac{1}{d_i^3}=\beta_1+\beta_2\frac{1}{d_i^3},\quad \beta_1=\frac{1}{\alpha},\quad \beta_2=\frac{\delta}{\alpha}.$$

Because the variability in Grouse consumption rate likely increases with the mean rate, suggesting a gamma distribution for the response, and completing the model specification.

As this third example illustrates, in the GLM setup we are not restricted to simple straight line forms and indeed can have any structure for the linear predictor that was possible for linear models.

Example 4: Linear models!

As already mentioned in the beginning of this section, any linear model is just a special case of a GLM. The link function is the 'identity' link and the response distribution is Gaussian.

11 Inference with GLMs

Inference with GLMs is based on the theory of maximum likelihood estimation. This section shows how this works in general for any GLM, from which emerges a nice link back to linear models.

11.1 The exponential family of distributions

The response variable in a GLM can have any distribution from the *exponential family*. A distribution belongs to the exponential family of distributions if its probability density function, or probability mass function, can be written as

$$f(y;\theta,\phi) = \exp\left[\{y\theta - b(\theta)\}/a(\phi) + c(y,\phi)\right], \tag{19}$$

where

- θ is know as the *canonical parameter* and is a function of the expectation of the response variable, $\mu \equiv \mathbb{E}(Y)$, and so θ is completely determined by $\boldsymbol{\beta}$.
- $\phi > 0$ is called the *dispersion parameter* and may or may not be known.
- $a(\cdot)$, $b(\cdot)$, and $c(\cdot)$ are arbitrary known functions that vary from one exponential family member to another.

As we will learn, if a distribution can be written in this manner, maximum likelihood estimation and inference are greatly simplified and can be handled in a unified framework.

To get a sense of how the exponential family works, let's work out the representation for three common distributions (Poisson, Gaussian, and Gamma).

Poisson distribution

The probability mass function of a Poisson distribution is

$$f(y;\mu) = \frac{e^{-\mu}\mu^y}{y!}, \quad y = 0,1,2,\ldots, \quad \mu > 0.$$

Let's first write $f(y;\mu)$ as $\exp\{\log f(y;\mu)\}$. We have that

$$\log f(y;\mu) = -\mu + y\log(\mu) - log(y!) \Rightarrow \exp\{\log f(y;\mu)\} = \exp\{-\mu + y\log(\mu) - log(y!)\}.$$

Now, let's rearrange the equation in the form of Equation (19):

$$\exp\{\log f(y;\mu)\} = \left\{\frac{y\log(\mu) - \mu}{1} + [-log(y!)]\right\}.$$

This is of the form of (19) with

- $\theta = \log\mu \Rightarrow \mu = e^\theta$
- $b(\theta) = \mu = e^\theta$
- $a(\phi) = \phi = 1$
- $c(y,\phi) = -log(y!)$

Note that $a(\phi) = \phi = 1$ since the Poisson distribution is a one parameter distribution.

Gaussian distribution

The probability density function of a Gaussian distribution is given by

$$f(y;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}(y-\mu)^2\right\}, \quad y \in \mathbb{R}, \quad \mu \in \mathbb{R}, \quad \sigma > 0.$$

As before, writing $f(y;\mu,\sigma^2)$ as $\exp\{\log f(y;\mu,\sigma^2)\}$:

$$\begin{aligned}\log f(y;\mu,\sigma^2) &= -\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}(y-\mu)^2 \\ &= -\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}(y^2 - 2\mu y + \mu^2),\end{aligned}$$

and thus

$$\begin{aligned}\exp\{\log f(y;\mu,\sigma^2)\} &= \exp\left\{-\frac{1}{2\sigma^2}(y^2 - 2\mu y + \mu^2) - \frac{1}{2}\log(2\pi\sigma^2)\right\} \\ &= \exp\left\{\frac{y\mu - \mu^2/2}{\sigma^2} - \frac{1}{2}\log(2\pi\sigma^2) - \frac{y^2}{2\sigma^2}\right\}\end{aligned}$$

This is of the form of (19) with

- $\theta = \mu$
- $b(\theta) = \frac{\mu^2}{2} = \frac{\theta^2}{2}$
- $a(\phi) = \phi = \sigma^2$
- $c(y,\phi) = -\frac{1}{2}\log(2\pi\sigma^2) - \frac{y^2}{2\sigma^2} = -\frac{1}{2}\log(2\pi\phi) - \frac{y^2}{2\phi}$

Gamma distribution

The density of the Gamma distribution is usually given by

$$f(y;\nu,\lambda) = \frac{\lambda^\nu}{\Gamma(\nu)} y^{\nu-1} e^{-\lambda y}, \quad y > 0, \quad \lambda > 0, \quad \nu > 0,$$

where ν describes the shape and λ the rate (inverse of a scale parameter) of the distribution. Under this parametrization, $E(Y) \equiv \mu = \frac{\nu}{\lambda}$. However, for the purposes of GLM, it is convenient to reparametrize the distribution by its mean, which can be achieved by letting $\lambda = \frac{\nu}{\mu}$, leading to

$$f(y;\nu,\mu) = \frac{1}{\Gamma(\nu)} \left(\frac{\nu}{\mu}\right)^\nu y^{\nu-1} e^{-\frac{\nu}{\mu}y}.$$

Note that now $E(Y) = \mu$. As for the other two examples,

$$\log f(y;\nu,\mu) = -\log[\Gamma(\nu)] + \nu\log(\nu) - \nu\log(\mu) + (\nu-1)\log(y) - \frac{\nu y}{\mu}.$$

Hence,

$$\begin{aligned}\exp\{\log f(y;\nu,\mu)\} &= \exp\left\{-\frac{\nu y}{\mu} - \nu\log(\mu) + (-\log[\Gamma(\nu)] + \nu\log(\nu) + (\nu-1)\log(y))\right\} \\ &= \exp\left\{\frac{y\,(-1/\mu) - \log(\mu)}{1/\nu} + (-\log[\Gamma(\nu)] + \nu\log(\nu) + (\nu-1)\log(y))\right\}.\end{aligned}$$

This is of the form of (19) with

- $\theta = -\frac{1}{\mu} \Rightarrow \mu = -\frac{1}{\theta}$
- $b(\theta) = \log(\mu) = \log(-\frac{1}{\theta}) = -\log(-\theta)$
- $a(\phi) = \phi = \frac{1}{\nu}$
- $c(y, \phi) = -\log[\Gamma(\nu)] + \nu \log(\nu) + (\nu - 1)\log(y) = -\log[\Gamma(1/\phi)y] + \frac{1}{\phi}\log(\frac{y}{\phi})$

In all these three cases, $a(\phi) = \phi$ and in fact, generally, we only need the case where a is a linear function so that $a(\phi) = \phi/\omega$, where ϕ is constant over all observations and ω is a known weight that varies from observation to observation. We will assume this going forward.

We now derive the mean and variance of the exponential family of distributions. To this end, note that the log likelihood for a single observation y is given by

$$l(\theta, \phi) = \log f(y; \theta, \phi) = \frac{\omega\{y\theta - b(\theta)\}}{\phi} + c(y, \phi).$$

Taking derivatives with respect to θ gives

$$\frac{\partial l}{\partial \theta} = \frac{\omega\{y - b'(\theta)\}}{\phi},$$

where prime denotes differentiation with respect to θ. Taking the expectation over y gives

$$\mathbb{E}\left[\frac{\partial l}{\partial \theta}\right] = \frac{\omega\{\mathbb{E}[Y] - b'(\theta)\}}{\phi}.$$

From general likelihood theory, it is known that

$$\mathbb{E}\left[\frac{\partial l}{\partial \theta}\right] = 0,$$

which hold under regularity conditions satisfied by the exponential family (basically that we can reverse the derivative and integral operations, meaning that the support of the response variable does not depend on θ). This fact implies that

$$\frac{\omega\{\mathbb{E}[Y] - b'(\theta)\}}{\phi} = 0 \Rightarrow E[Y] \equiv \mu = b'(\theta). \tag{20}$$

Note that $b'(\theta)$ is one-to-one, so there is a one to one correspondence between μ and θ. The result in (20) is the key to linking the GLM model parameter $\boldsymbol{\beta}$ to the canonical parameter of the exponential family θ. In a GLM, $\boldsymbol{\beta}$ determines the mean of the response variable and via, Equation (20), thereby determines the canonical parameter for each response observation.

Now taking second derivatives leads to

$$\frac{\partial^2 l}{\partial \theta^2} = \frac{-\omega b''(\theta)}{\phi}.$$

General likelihood theory also tells us that

$$\mathbb{E}\left[\frac{\partial^2 l}{\partial \theta^2}\right] = -\mathbb{E}\left[\left(\frac{\partial l}{\partial \theta}\right)^2\right]$$

First note that

$$\mathbb{E}\left[\frac{\partial^2 l}{\partial \theta^2}\right] = -\frac{\omega b''(\theta)}{\phi}.$$

Further note that

$$\left(\frac{\partial l}{\partial \theta}\right)^2 = \frac{\omega^2\{y - b'(\theta)\}^2}{\phi^2} = \frac{\omega^2(y - \mu)^2}{\phi^2} \Rightarrow \mathbb{E}\left[\left(\frac{\partial l}{\partial \theta}\right)^2\right] = \frac{\omega^2}{\phi^2}\mathbb{E}[(Y - \mu)^2] = \frac{\omega^2}{\phi^2}\text{var}(Y).$$

Therefore, it holds that

$$-\frac{\omega b''(\theta)}{\phi} = -\frac{\omega^2}{\phi^2}\text{var}(Y) \Rightarrow \text{var}(Y) = \frac{\phi b''(\theta)}{\omega} = b''(\theta)a(\phi).$$

So, we see that the mean is a function of θ only, while the variance is a product of functions of the location and scale. Given the one-to-one correspondence between μ and θ we can also define the *variance function* $V(\mu) = b''(\theta)/\omega$, and write $\text{var}(Y) = V(\mu)\phi$. The variance function $V(\mu)$ describes how the variance relates to the mean through Equation (20). In the Gaussian case we had $\omega = 1$ and $b(\theta) = \theta^2$ implying that $V(\mu) = b''(\theta) = 1$ and so the variance is independent of the mean. For other distributions this is not true, making the Gaussian case exceptional. Further observe that $\text{var}(Y) = 1 \times \phi = \sigma^2$, as we already knew. In turn, for the Poisson distribution, we had $\omega = 1$ and $b(\theta) = e^\theta$ implying that $b''(\theta) = e^\theta = \mu$ and thus $V(\mu) = \mu$, i.e., the variance increases with the mean. Also, $\text{var}(Y) = \phi \times \mu = \mu$ as it should be (it is widely known that for the Poisson distribution the mean coincides with the variance). Finally for the Gamma distribution we had that $\omega = 1$ and $b(\theta) = -\log(-\theta) \Rightarrow b''(\theta) = 1/\theta^2$ and thus $V(\mu) = \mu^2$, that is, the variance also increases with the mean for the Gamma distribution. Lastly, $\text{var}(Y) = b''(\theta)\phi = \mu^2/\nu$.

Link function

As it has been mentioned at the beginning of section 10, the *link function* of a GLM connects the random component and the linear predictor. That is, a GLM states that the linear predictor for observation i, $\eta_i = \mathbf{X}_i\beta = \sum_{j=1}^p \beta_j X_{ij}$ (if the model contains an intercept then $X_{i1} = 1$) relates to μ_i by $\eta_i = g(\mu_i)$, for a link function g. Equivalently, $\mu_i = g^{-1}(\eta_i)$, i.e., the function g^{-1} maps linear predictor values to the mean.

In principle, any smooth monotonic function g will do, but there are some convenient and common choices for the standard GLMs.

In the Gaussian linear model, the identity link, $\eta_i = g(\mu_i) = \mu_i$, is the obvious selection, but another choice would give $y_i = g^{-1}(\mathbf{X}_i\boldsymbol{\beta}) + \epsilon_i$.

For the Poisson GLM, the mean μ_i must be positive so $\eta_i = \mu_i$ will not work conveniently since η_i can be negative. The standard choice is $\mu_i = e^{\eta_i}$ so that $\eta_i = \log \mu_i$ which ensures $\mu_i > 0$. This log link means that additive effects of the predictors lead to multiplicative effects on the mean.

The link function g that transforms the mean μ_i to the canonical parameter θ_i in (19) is called the *canonical link function*, i.e., g is such that $\eta_i = g(\mu_i) = \theta_i$. The canonical link for the Gaussian distribution is $g(\mu) = \mu$ and for the Poisson distribution it is $g(\mu) = \log \mu$. Constants in θ are generally omitted from the canonical link. For example, for the Gamma distribution $\theta = -1/\mu$ and the canonical link is $g(\mu) = 1/\mu$. When g is the canonical link function, the derivation of the maximum likelihood estimator of $\boldsymbol{\beta}$ is simplified as we shall see just next. Nevertheless, even in this case, the estimating equations (except in the case of the Gaussian distribution with identity link) are nonlinear functions of the regression parameters $\boldsymbol{\beta}$ and generally still require iterative methods for their solution, so we shall proceed with the general case where g does not necessarily need to be the canonical link function.

11.2 GLM fitting

Recall that a GLM models an $n \times 1$ vector of independent response variables $\mathbf{y}$, where $\boldsymbol{\mu} \equiv \mathbb{E}(\mathbf{y})$, via

$$g(\mu_i) = \mathbf{X}_i\boldsymbol{\beta}, \quad \text{and} \quad y_i \sim f(y_i; \theta_i, \phi),$$

where $f(y_i; \theta_i, \phi)$ indicates an exponential family of distributions with canonical parameter θ_i, which is determined by μ_i via Equation (20) and hence ultimately by $\boldsymbol{\beta}$. Because the response variables are mutually independent, the likelihood of $\boldsymbol{\beta}$ is

$$L(\boldsymbol{\beta}) = \prod_{i=1}^n f(y_i; \theta_i, \phi),$$

and hence the log likelihood of $\boldsymbol{\beta}$ is

$$l(\boldsymbol{\beta}) = \sum_{i=1}^n l_i(\boldsymbol{\beta}) = \sum_{i=1}^n \log f(y_i; \theta_i, \phi) = \sum_{i=1}^n \left\{ \frac{\omega_i[y_i\theta_i - b(\theta_i)]}{\phi} + c(y_i, \phi) \right\},$$

where the dependence of the two last terms on the right hand side on $\boldsymbol{\beta}$ is through the dependence of the θ_i on $\boldsymbol{\beta}$. Finding the maximum likelihood estimate of $\boldsymbol{\beta}$, which as before we assume it is an $p \times 1$ dimensional vector, requires the score vector, i.e., the vector formed by the partial derivatives of $l(\boldsymbol{\beta})$ with respect to each β_j, for $j = 1, \ldots, p$, that is,

$$\mathbf{U}(\boldsymbol{\beta}) = \begin{bmatrix} \frac{\partial l(\boldsymbol{\beta})}{\partial \beta_1} \\ \vdots \\ \frac{\partial l(\boldsymbol{\beta})}{\partial \beta_p} \end{bmatrix}$$

We have that

$$\frac{\partial l(\boldsymbol{\beta})}{\partial \beta_j} = \sum_{i=1}^{n} \frac{\partial}{\partial \beta_j} l_i(\boldsymbol{\beta}), \quad \text{for } j = 1, \ldots, p.$$

To differentiate the log likelihood we use the chain rule,

$$\frac{\partial l_i(\boldsymbol{\beta})}{\partial \beta_j} = \frac{\partial l_i}{\partial \theta_i} \frac{\mathrm{d}\theta_i}{\mathrm{d}\mu_i} \frac{\mathrm{d}\mu_i}{\mathrm{d}\eta_i} \frac{\partial \eta_i}{\partial \beta_j}. \tag{21}$$

Note that if the link function g is the canonical link function (and so $\eta_i = g(\mu_i) = \theta_i$) the chain rule above simplifies to

$$\frac{\partial l_i(\boldsymbol{\beta})}{\partial \beta_j} = \frac{\partial l_i}{\partial \theta_i} \frac{\partial \theta_i}{\partial \beta_j}.$$

We shall proceed with the most general case where g does not necessarily need to be the canonical link function. Let's then workout each derivative on the right-hand side of (21). We have that

$$\begin{aligned} \frac{\partial l_i}{\partial \theta_i} &= \frac{\partial}{\theta_i} \left\{ \frac{\omega_i}{\phi} [y_i \theta_i - b(\theta_i)] + c(y_i, \phi) \right\} \\ &= \frac{\omega_i}{\phi} [y_i - b'(\theta_i)] \\ &= \frac{\omega_i}{\phi} (y_i - \mu_i), \quad \mu_i = \mathbb{E}(y_i) = b'(\theta_i). \end{aligned}$$

For determining $\frac{\mathrm{d}\theta_i}{\mathrm{d}\mu_i}$, we note that from Equation (20) we have that $\mu_i = b'(\theta_i)$, and so

$$\frac{\mathrm{d}\mu_i}{\mathrm{d}\theta_i} = b''(\theta_i),$$

and because b' is a one to one function, one must has

$$\frac{\mathrm{d}\theta_i}{\mathrm{d}\mu_i} = \frac{1}{\frac{\mathrm{d}\mu_i}{\mathrm{d}\theta_i}} = \frac{1}{b''(\theta_i)}.$$

Similarly, and noting that $g(\mu_i) = \eta_i$, we have that

$$\frac{\mathrm{d}\eta_i}{\mathrm{d}\mu_i} = g'(\mu_i),$$

and because g is smooth and monotonic then

$$\frac{\mathrm{d}\mu_i}{\mathrm{d}\eta_i} = \frac{1}{\frac{\mathrm{d}\eta_i}{\mathrm{d}\mu_i}} = \frac{1}{g'(\mu_i)}.$$

Finally, note that

$$\eta_i = \mathbf{X}_i \boldsymbol{\beta} = \sum_{j=1}^{p} X_{ij} \beta_j \Rightarrow \frac{\partial \eta_i}{\partial \beta_j} = X_{ij},$$

where we assume that $X_{i1} = 1$ if the model has an intercept. Putting all terms together, leads to

$$\frac{\partial l_i(\boldsymbol{\beta})}{\partial \beta_j} = \frac{\omega_i}{\phi}(y_i - \mu_i) \times \frac{1}{b''(\theta_i)} \times \frac{1}{g'(\mu_i)} \times X_{ij},$$

and hence

$$\begin{aligned}\frac{\partial}{\partial \beta_j} l(\boldsymbol{\beta}) &= \frac{1}{\phi} \sum_{i=1}^{n} \frac{(y_i - \mu_i) X_{ij}}{\frac{b''(\theta_i)}{\omega_i} g'(\mu_i)} \\ &= \frac{1}{\phi} \sum_{i=1}^{n} \frac{(y_i - \mu_i) X_{ij}}{V(\mu_i) g'(\mu_i)}, \quad j = 1, \ldots, p,\end{aligned}$$

recalling that $V(\mu_i) = b''(\theta_i)/\omega_i$. To maximize $l(\boldsymbol{\beta})$, we find the values $\hat{\boldsymbol{\beta}}$ for which these derivatives are 0. Note that $\hat{\boldsymbol{\beta}}$ will not depend on ϕ. These equations do not have an analytic solution and so we resort to a numerical procedure and in order to do that we require the second derivatives of the log likelihood. We need to find

$$\begin{aligned}\frac{\partial^2 l(\boldsymbol{\beta})}{\partial \beta_k \beta_j} &= \frac{\partial}{\partial \beta_k} \left(\frac{\partial l(\boldsymbol{\beta})}{\partial \beta_j} \right) \\ &= \frac{\partial}{\partial \beta_k} \left(\sum_{i=1}^{n} \frac{\partial}{\partial \beta_j} l_i(\boldsymbol{\beta}) \right) \\ &= \sum_{i=1}^{n} \frac{\partial}{\partial \beta_k} \left(\frac{\partial}{\partial \beta_j} l_i(\boldsymbol{\beta}) \right).\end{aligned}$$

We have that

$$\frac{\partial}{\partial \beta_k} \left(\frac{\partial}{\partial \beta_j} l_i(\boldsymbol{\beta}) \right) = \frac{\partial}{\partial \beta_k} \left(\frac{1}{\phi} \frac{(y_i - \mu_i) X_{ij}}{V(\mu_i) g'(\mu_i)} \right) = \frac{1}{\phi} \frac{\partial}{\partial \beta_k} \left(\frac{(y_i - \mu_i) X_{ij}}{V(\mu_i) g'(\mu_i)} \right)$$

We will be using the chain rule again as follows

$$\frac{\mathrm{d}\left(\frac{\partial}{\partial \beta_j} l_i(\boldsymbol{\beta}) \right)}{\mathrm{d}\mu_i} \frac{\mathrm{d}\mu_i}{\mathrm{d}\eta_i} \frac{\partial \eta_i}{\partial \beta_k}.$$

As before,

$$\frac{\mathrm{d}\mu_i}{\mathrm{d}\eta_i} = \frac{1}{g'(\mu_i)}, \quad \frac{\partial \eta_i}{\partial \beta_k} = X_{ik}.$$

For the first term we have that

$$\frac{\mathrm{d}\left(\frac{\partial}{\partial \beta_j} l_i(\boldsymbol{\beta}) \right)}{\mathrm{d}\mu_i} = \frac{\mathrm{d}}{\mathrm{d}\mu_i} \left((y_i - \mu_i) X_{ij} \times \frac{1}{V(\mu_i) g'(\mu_i)} \right) = -X_{ij} \frac{1}{V(\mu_i) g'(\mu_i)} + \frac{\mathrm{d}}{\mathrm{d}\mu_i} \left(\frac{1}{V(\mu_i) g'(\mu_i)} \right) (y_i - \mu_i) X_{ij}$$

$$\frac{\mathrm{d}}{\mathrm{d}\mu_i} \left(\frac{1}{V(\mu_i) g'(\mu_i)} \right) = -\frac{V'(\mu_i) g'(\mu_i) + V(\mu_i) g''(\mu_i)}{g'(\mu_i)^2 V(\mu_i)^2}$$

and so

$$\frac{\mathrm{d}\left(\frac{\partial}{\partial \beta_j} l_i(\boldsymbol{\beta}) \right)}{\mathrm{d}\mu_i} = -X_{ij} \frac{1}{V(\mu_i) g'(\mu_i)} - \frac{V'(\mu_i) g'(\mu_i) + V(\mu_i) g''(\mu_i)}{g'(\mu_i)^2 V(\mu_i)^2} (y_i - \mu_i) X_{ij}$$

Putting all terms together leads to

$$\begin{aligned}\frac{\partial^2 l(\boldsymbol{\beta})}{\partial \beta_k \beta_j} &= -\frac{1}{\phi} \sum_{i=1}^{n} \left\{ \left[X_{ij} \frac{1}{V(\mu_i) g'(\mu_i)} + \frac{V'(\mu_i) g'(\mu_i) + V(\mu_i) g''(\mu_i)}{g'(\mu_i)^2 V(\mu_i)^2} (y_i - \mu_i) X_{ij} \right] \frac{1}{g'(\mu_i)} X_{ik} \right\} \\ &= -\frac{1}{\phi} \sum_{i=1}^{n} \left\{ \frac{X_{ik} X_{ij}}{g'(\mu_i)^2 V(\mu_i)} + \frac{(y_i - \mu_i) X_{ij} X_{ik}}{g'(\mu_i)^2 V(\mu_i)} \left(\frac{g''(\mu_i)}{g'(\mu_i)} + \frac{V'(\mu_i)}{V(\mu_i)} \right) \right\}\end{aligned}$$

Note that

$$\mathbb{E}\left(\frac{\partial^2 l}{\partial\beta_k\partial\beta_j}\right) = -\frac{1}{\phi}\sum_{i=1}^{n}\frac{X_{ik}X_{ij}}{g'(\mu_i)^2V(\mu_i)},$$

since $\mathbb{E}(y_i - \mu_i) = \mathbb{E}(y_i) - \mu_i = \mu_i - \mu_i = 0$.

Defining $w_i^{-1} = g'(\mu_i)^2V(\mu_i)$ and $\mathbf{W} = \text{diag}(w_i)$, we therefore have that $\boldsymbol{\mathcal{I}} = -\mathbb{E}(\partial^2 l/\partial\boldsymbol{\beta}\partial\boldsymbol{\beta}^\mathsf{T}) = \mathbf{X}^\mathsf{T}\mathbf{W}\mathbf{X}/\phi$. From the general properties of maximum likelihood estimators we have that, in the large-sample limit,

$$\hat{\boldsymbol{\beta}} \sim N(\boldsymbol{\beta}, \boldsymbol{\mathcal{I}}^{-1}), \quad \boldsymbol{\mathcal{I}} = \mathbf{X}^\mathsf{T}\mathbf{W}\mathbf{X}/\phi. \tag{22}$$

For distributions with known scale parameter ϕ (e.g., the Poisson and binomial distributions) this result can be used to directly find confidence intervals for the parameters, but if the scale parameter is unknown (as it is the case for, e.g., the normal and gamma distributions), then it must be estimated (see section 11.3) and intervals must be based on an appropriate t distribution.

Applying Newton's method to find $\hat{\boldsymbol{\beta}}$ amounts to iterating (from some initial guess $\boldsymbol{\beta}^{(0)}$)

$$\boldsymbol{\beta}^{(t+1)} = \boldsymbol{\beta}^{(t)} - \mathbf{H}(\boldsymbol{\beta}^{(t)})^{-1}\mathbf{U}(\boldsymbol{\beta}^{(t)}), \quad \mathbf{H}(\boldsymbol{\beta}) = \frac{\partial^2}{\partial\boldsymbol{\beta}\boldsymbol{\beta}^\mathsf{T}}l(\boldsymbol{\beta}),$$

until successive $\boldsymbol{\beta}^{(t)}$ do not change. A useful feature of Newton's method is that it converges just as well with the second derivative matrix replaced by its expectation. This modification is known as the *Fisher scoring* method, whose updating equations are as follows:

$$\begin{aligned}\boldsymbol{\beta}^{(t+1)} &= \boldsymbol{\beta}^{(t)} - (\mathbb{E}[\mathbf{H}(\boldsymbol{\beta}^{(t)})])^{-1}\mathbf{U}(\boldsymbol{\beta}^{(t)}) \\ &= \boldsymbol{\beta}^{(t)} + \boldsymbol{\mathcal{I}}(\boldsymbol{\beta}^{(t)})^{-1}\mathbf{U}(\boldsymbol{\beta}^{(t)}).\end{aligned}$$

Note that

$$\begin{aligned}\boldsymbol{\mathcal{I}}(\boldsymbol{\beta}^{(t)})^{-1} &= (\mathbf{X}^\mathsf{T}\mathbf{W}^{(t)}\mathbf{X})^{-1}\phi \\ \mathbf{U}(\boldsymbol{\beta}^{(t)}) &= \frac{1}{\phi}\mathbf{X}^\mathsf{T}\mathbf{W}^{(t)}\mathbf{G}^{(t)}(\mathbf{y} - \boldsymbol{\mu}^{(t)}), \quad \mathbf{G} = \text{diag}\{g'(\mu_i)\},\end{aligned}$$

and so the updating equations of the Fisher scoring algorithm are given by

$$\begin{aligned}\boldsymbol{\beta}^{(t+1)} &= \boldsymbol{\beta}^{(t)} + (\mathbf{X}^\mathsf{T}\mathbf{W}^{(t)}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{W}^{(t)}\mathbf{G}^{(t)}(\mathbf{y} - \boldsymbol{\mu}^{(t)}) \\ &= (\mathbf{X}^\mathsf{T}\mathbf{W}^{(t)}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{W}^{(t)}\{\mathbf{G}^{(t)}(\mathbf{y} - \boldsymbol{\mu}^{(t)}) + \mathbf{X}\boldsymbol{\beta}^{(t)}\} \\ &= (\mathbf{X}^\mathsf{T}\mathbf{W}^{(t)}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{W}^{(t)}\mathbf{z}^{(t)}\end{aligned}$$

where $z_i^{(t)} = g'(\mu_i^{(t)})(y_i - \mu_i^{(t)}) + \eta_i^{(t)}$ and $\eta_i^{(t)} = \mathbf{X}_i\boldsymbol{\beta}^{(t)}$. So $\boldsymbol{\beta}^{(t+1)}$ is recognisable as the minimizer of the weighted sum of squares, $\sum_i w_i(z_i - \mathbf{X}_i\boldsymbol{\beta})^2$.

Hence maximum likelihood estimation for a GLM is achieved using the *iteratively re-weighted least squares* (IRLS) algorithm.

1. Set $t = 0$, $\mu_i^{(0)} = y_i + \Delta_i$ and $\eta_i^{(0)} = g(\mu_i^{(0)})$, where $\Delta_i = 0$ usually, but may be a small constant if needed to ensure that $\eta_i^{(0)}$ is well defined and finite (for instance for count data, Poisson regression with a log link, if some counts are zero). Then iterate the following two steps until convergence.
2. Form $z_i^{(t)} = g'(\mu_i^{(t)})(y_i - \mu_i^{(t)}) + \eta_i^{(t)}$ and $w_i^{(t)} = \{g'(\mu_i^{(t)})^2V(\mu_i^{(t)})\}^{-1}$.
3. Find $\boldsymbol{\beta}^{(t+1)} = \text{argmin}_{\boldsymbol{\beta}}\sum_i w_i^{(t)}(z_i^{(t)} - \mathbf{X}_i\boldsymbol{\beta})^2$, by standard linear model methods. Increment t.

Weighted least squares: a brief note

Recall that an expression for the value of $\boldsymbol{\beta}$ minimizing

$$\sum_{i=1}^{n}(z_i - \mathbf{X}_i\boldsymbol{\beta})^2,$$

is $\hat{\boldsymbol{\beta}} = (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{z}$ (although as we know this is not the expression to use, in general, for actual computation). Let us verify that $\hat{\boldsymbol{\beta}} = (\mathbf{X}^\mathsf{T}\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{W}\mathbf{z}$ is indeed the minimizer of the weighted sum of squares

$$\sum_{i=1}^{n} w_i(z_i - \mathbf{X}_i\boldsymbol{\beta})^2. \tag{23}$$

To this end let $\tilde{\mathbf{W}}$ be the diagonal matrix such that $\tilde{W}_{ii} = \sqrt{w_i}$, and let $\mathbf{W} = \tilde{\mathbf{W}}\tilde{\mathbf{W}}$, i.e. the diagonal matrix such that $W_{ii} = w_i$.

$$\sum_{i=1}^{n} w_i(z_i - \mathbf{X}_i\boldsymbol{\beta})^2 = \sum_{i=1}^{n}\{\sqrt{w_i}(z_i - \mathbf{X}_i\boldsymbol{\beta})\}^2 = \sum_{i=1}^{n}(\sqrt{w_i}z_i - \sqrt{w_i}\mathbf{X}_i\boldsymbol{\beta})^2 = \sum_{i=1}^{n}(\tilde{z}_i - \tilde{\mathbf{X}}_i\boldsymbol{\beta})^2$$

where $\tilde{\mathbf{z}} = \tilde{\mathbf{W}}\mathbf{z}$ and $\tilde{\mathbf{X}} = \tilde{\mathbf{W}}\mathbf{X}$. So using the formal expression for a least squares estimate of $\boldsymbol{\beta}$ gives

$$\hat{\boldsymbol{\beta}} = (\tilde{\mathbf{X}}^\mathsf{T}\tilde{\mathbf{X}})^{-1}\tilde{\mathbf{X}}^\mathsf{T}\tilde{\mathbf{z}} = (\mathbf{X}^\mathsf{T}\tilde{\mathbf{W}}\tilde{\mathbf{W}}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\tilde{\mathbf{W}}\tilde{\mathbf{W}}\mathbf{z} = (\mathbf{X}^\mathsf{T}\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{W}\mathbf{z}.$$

Example

Let us revisit the AIDS data example from section 10. Recall that after writing the model in a GLM form we have

$$\log(\mu_i) = \beta_1 + \beta_2 t_i.$$

As a starting point it makes sense to try a Poisson distribution with a log link (*Poisson regression*). We now implement the IRLS algorithm. We need $g'(\mu) = (\log\mu)' = \frac{1}{\mu}$. We further have that for the Poisson distribution $V(\mu) = \mu$. Note that at any iteration t we have

$$z_i^{(t)} = g'(\mu_i^{(t)})(y_i - \mu_i^{(t)}) + \eta_i^{(t)} = \frac{1}{\mu_i^{(t)}}(y_i - \mu_i^{(t)}) + \eta_i^{(t)} = \frac{y_i}{\mu_i^{(t)}} - 1 + \eta_i^{(t)},$$

$$w_i^{(t)} = \{g'(\mu_i^{(t)})^2 V(\mu_i^{(t)})\}^{-1} = \left(\frac{1}{{\mu_i^{(t)}}^2}\mu_i^{(t)}\right)^{-1} = \mu_i^{(t)}.$$

```
y <- c(12,14,33,50,67,74,123,141,165,204,253,246,240)
t <- 1:13
mu <- y; eta <- log(mu) ## initialize mean and linear predictor
converged <- FALSE
while (!converged) {
  z <- y/mu - 1 + eta ## create pseudodata
  w <- mu             ## create iterative weights
  fit <- lm(z ~ t, weights = w) ## fit working model
  eta.old <- eta      ## store old linear predictor
  eta <- fitted(fit)  ## get new linear predictor (X \beta)
  mu <- exp(eta)      ## get new mean
  ## test for convergence ...
  if (max(abs(eta-eta.old))<1e-6*mean(abs(eta))) converged <- TRUE
}
coef(fit)

## (Intercept)           t
##   3.1405895   0.2021212
```

Note that we have used the `lm` function with an argument new to us: `weights`. By typing `help(lm)` the following can be read for this argument:

"an optional vector of weights to be used in the fitting process. (...) If non-NULL, weighted least squares is used with weights `weights` (that is, minimizing $sum(w * e^2)$) (...)"

This is exactly what we want in (23). It is also easy to implement the IRWLS without using the `lm` function. Of course, we know that computing the β estimates, at each iteration, this way (through the use of the `solve` function) may be unstable in certain situations.

```
mu <- y; eta <- log(mu)
t_vec <-cbind(rep(1, length(t)), t)
converged <- FALSE
iter <- 0 #counts how many iterations until convergence
while (!converged) {
  z <- y/mu - 1 + eta
  w <- as.vector(mu)
  coefs <- solve(t(t_vec)%*%diag(w)%*%t_vec)%*%t(t_vec)%*%diag(w)%*%cbind(z)
  eta.old <- eta
  eta <- t_vec%*%coefs
  mu <- exp(eta)
  if (max(abs(eta-eta.old))<1e-6*mean(abs(eta))) converged <- TRUE
  iter <- iter + 1
}
as.numeric(coefs)

## [1] 3.1405895 0.2021212

iter

## [1] 4
```

We now compare with the results obtained using the `glm` function (which we will be using extensively in the next section).

```
mod1 <- glm(y ~ t, family = poisson)
summary(mod1)

##
## Call:
## glm(formula = y ~ t, family = poisson)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) 3.140590   0.078247   40.14   <2e-16 ***
## t           0.202121   0.007771   26.01   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 872.206  on 12  degrees of freedom
## Residual deviance:  80.686  on 11  degrees of freedom
## AIC: 166.37
##
## Number of Fisher Scoring iterations: 4
```

11.3 GLM checking and inference

Model checking for GLMs is performed using residuals, in the same way as for linear models. However, in the case of GLMs standardization of the residuals is particularly necessary. For GLMs, the main reason for not simply examining the raw residuals, $\hat{\epsilon}_i = y_i - \hat{\mu}_i$, is the difficulty of checking the validity of the assumed mean variance relationship from the raw residuals. For example, if a Poisson model is used, then the variance of the residuals should increase in direct proportion to the size of the fitted values, $\hat{\mu}_i$. However, if raw residuals are plotted against fitted values, it is very difficult to judge whether the residual variability is increasing in proportion to the mean, as opposed to, say, the square root or square of the mean. For this reason it is usual to standardize GLM residuals so that, if the model assumptions are correct, the standardized residuals should have approximately equal variance and behave, as far as possible, like residuals from a linear model.

The most obvious way to standardize the residuals is to divide them by a quantity proportional to their standard deviation according to the fitted model. This gives rise to the *Pearson residuals*, defined as

$$\hat{\epsilon}_i = \frac{y_i - \hat{\mu}_i}{\sqrt{V(\hat{\mu}_i)}},$$

which should approximately have zero mean and variance ϕ, if the model is correct. These residuals should not display any trend when plotted against the fitted values, or any predictors (whether included in the model or not), if the model is correctly specified. Further, if the spread of the residuals is not constant across the fitted values, then this suggests that the assumed mean-variance relationship is not correct.

As already mentioned, when ϕ is known (22) can be used directly for finding confidence intervals for β_is or testing hypotheses about them. When ϕ is unknown then an estimator is needed and although it is in principle possible to estimate ϕ by maximum likelihood as well, this is rarely done. Instead, a possible estimator is based on the sum of squares of the Pearson residuals, resulting in the so-called Pearson estimator,

$$\hat{\phi} = \frac{1}{n-p} \sum_{i=1}^{n} \frac{(y_i - \hat{\mu}_i)^2}{V(\hat{\mu}_i)}, \tag{24}$$

where p is the number of parameters (dimension of $\boldsymbol{\beta}$). This can then be plugged into (22).

We now introduce the concept of *deviance* which is a key one in generalized linear models and that plays a similar role to the residual sum of squares in a linear model. Intuitively, it measures the deviance of the fitted generalized linear model with respect to a perfect model for the sample $\{(y_i, \mathbf{x}_i)\}_{i=1}^n$. This perfect model is known as the *saturated model*: the model with one parameter per data point (and thus n parameters) and perfect fit, i.e., $\hat{\boldsymbol{\mu}} = \mathbf{y}$. A perfect fit sounds good but the saturated model is uninformative because it does not summarize the data but merely repeats them in full and it thus does not have the advantages that a simpler model has because of its parsimony, such as better estimation of the true relation between the response variable and predictors. However, the full model gives a baseline for measuring the discrepancy for an intermediate model with, say, p parameters. Let $l(\hat{\boldsymbol{\beta}})$ and l_S denote the maximized log likelihood for the model under consideration and for the saturated model, respectively. Note that l_S is the highest value that the log likelihood could possibly have, given the data. Formally, the deviance is defined as

$$D = 2\{l_S - l(\hat{\boldsymbol{\beta}})\}\phi \tag{25}$$

The log likelihood $l(\hat{\boldsymbol{\beta}})$ is always smaller than l_S. As a consequence, the deviance is always equal or greater than zero. The greater the deviance the poorer the fit or, equivalently stated, the deviance gets smaller the more closely the model fits the data. For the fitted model under consideration, let $\hat{\theta}_i$ denote the estimate of the canonical parameter θ_i, corresponding to the estimated mean $\hat{\mu}_i$. Similarly, let $\tilde{\theta}_i$ denote the estimate of θ_i for the saturated model, with corresponding mean given by $\tilde{\mu}_i = y_i$. Provided that the response observations are independent and for an exponential family distribution, with $a_i(\phi) = \phi/\omega_i$, the deviance D in (25) simplifies to

$$\begin{aligned} D &= 2\sum_{i=1}^{n} \left\{ \frac{\omega_i[y_i\tilde{\theta}_i - b(\tilde{\theta}_i)]}{\phi} + c(y_i, \phi) - \frac{\omega_i[y_i\hat{\theta}_i - b(\hat{\theta}_i)]}{\phi} - c(y_i, \phi) \right\} \phi \\ &= \sum_{i=1}^{n} 2\omega_i\{y_i(\tilde{\theta}_i - \hat{\theta}_i) - b(\tilde{\theta}_i) + b(\hat{\theta}_i)\}. \end{aligned} \tag{26}$$

Notice how the deviance does not depend on ϕ, which cancels.

For a Poisson GLM, from section 11.1, we have that $\hat{\theta}_i = \log(\hat{\mu}_i)$ and $b(\hat{\theta}_i) = e^{\hat{\theta}_i} = \hat{\mu}_i$. Similarly, for the saturated model, for which $\tilde{\mu}_i = y_i$, we have that $\tilde{\theta}_i = \log(y_i)$ and $b(\tilde{\theta}_i) = y_i$. Also, $\omega_i = 1$, for all $i = 1, \ldots, n$. The deviance is then equal to

$$\begin{aligned} D &= \sum_{i=1}^{n} 2\{y_i(\log y_i - \log \hat{\mu}_i) - y_i + \hat{\mu}_i\} \\ &= \sum_{i=1}^{n} 2\left\{y_i \log\left(\frac{y_i}{\hat{\mu}_i}\right) - y_i + \hat{\mu}_i\right\}. \end{aligned}$$

For a normal GLM, from section 11.1, we have that $\hat{\theta}_i = \hat{\mu}_i$ and $b(\hat{\theta}_i) = \frac{\hat{\theta}_i^2}{2} = \frac{\hat{\mu}_i^2}{2}$. Similarly, $\tilde{\theta}_i = y_i$ and $b(\tilde{\theta}_i) = \frac{y_i^2}{2}$ for the saturated model. We further have $\omega_i = 1, \forall i$. So the deviance equals to

$$\begin{aligned} D &= \sum_{i=1}^{n} 2\left\{y_i(y_i - \hat{\mu}_i) - \frac{y_i^2}{2} + \frac{\hat{\mu}_i^2}{2}\right\} \\ &= \sum_{i=1}^{n} (y_i^2 - 2y_i\hat{\mu}_i + \hat{\mu}_i^2) \\ &= \sum_{i=1}^{n} (y_i - \hat{\mu}_i)^2. \end{aligned} \tag{27}$$

For ordinary linear models this is the residual sum of squares: $\|\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}\|^2 = \|\mathbf{y} - \hat{\boldsymbol{\mu}}\|^2$ (in the notation of section 4).

A quantity related to the deviance is the *scaled deviance*, given by

$$D^* = \frac{D}{\phi} = 2\{l_S - l(\hat{\boldsymbol{\beta}})\}.$$

If $\phi = 1$, such as for the Poisson distribution, the deviance and the scaled deviance are the same. By the generalized likelihood ratio test in Equation (17) in section 9, we might expect that, if the model is correct, then approximately

$$D^* \sim \chi^2_{n-p}, \tag{28}$$

in the large sample limit. Note that in (17), r denotes the difference in the number of parameters in the two models being compared. Recall that the saturated model has n parameters and assuming that the model of interest has p parameters, then the difference in the number of parameters of the saturated model and the model being fitted is $n - p$, and hence this corresponds to the number of degrees of freedom of the limiting chi-squared distribution. Technically, the limiting argument justifying this large sample approximation in (28) is bogus, since it relies on the number of parameters in the model staying fixed, while the sample size tends to infinity, but the saturated model has as many parameters as data points. Asymptotic results are available for some of the distributions in the exponential family, to justify (28), as a large sample approximation under many circumstances, and it is exact for the normal case. In this case, however, similar to the case of a Gamma or an Inverse Gaussian distribution (see workshop 3), we do not know the dispersion parameter ϕ and so this test usually cannot be used directly as a goodness of fit statistic. For the binomial and Poisson distributions, $\phi = 1$, and so the test is practical. However, the accuracy of the approximation is dubious for small datasets. In the case of the binomial distribution with binary data, the approximation completely breaks down for reasons that are not discussed here.

If we want to compare a GLM model M_0, with p_0 predictors, that is nested within a GLM model M_1, with p_1 predictors (and so $p_0 < p_1$) using the GLRT in Equation (17), then the test statistic can be re-written in terms of D_0^* and D_1^* to obtain the approximate result that under the null hypothesis that the simpler model, M_0, is correct,

$$\begin{aligned} D_0^* - D_1^* &= 2\{l_S - l(\hat{\boldsymbol{\beta}}_\mathbf{0})\} - 2\{l_S - l(\hat{\boldsymbol{\beta}}_\mathbf{1})\} \\ &= 2\{l(\hat{\boldsymbol{\beta}}_\mathbf{1}) - l(\hat{\boldsymbol{\beta}}_\mathbf{0})\} \sim \chi^2_{p_1 - p_0}, \end{aligned}$$

where $l(\hat{\boldsymbol{\beta}}_{\mathbf{1}})$ and $l(\hat{\boldsymbol{\beta}}_{\mathbf{0}})$ are the maximized log likelihoods under model M_1 and M_0, respectively. This is only useful if the scale parameter ϕ is known so that D^* can be calculated. Note that the scaled deviance D^* apparently removes the effect of ϕ but it is, in fact, still dependent on ϕ, since this is hidden in the likelihood (see Equation (26)). Of course, we can use this result with $\hat{\phi}$ plugged into D_j^* to obtain an estimate $\hat{D}_j^*$, $j \in \{0, 1\}$, but a slightly better approximation is the one presented next. Under H_0, the two following approximate results hold

$$D_0^* - D_1^* \sim \chi^2_{p_1 - p_0} \quad \text{and} \quad D_1^* \sim \chi^2_{n-p_1},$$

and, if $D_0^* - D_1^*$ and D_1^* are treated as asymptotically independent, this implies that

$$\begin{aligned} F &= \frac{(D_0^* - D_1^*)/(p_1 - p_0)}{D_1^*/(n - p_1)} \\ &= \frac{(D_0 - D_1)/\phi/(p_1 - p_0)}{D_1/\phi/(n - p_1)} \\ &= \frac{(D_0 - D_1)/(p_1 - p_0)}{D_1/(n - p_1)} \sim F_{p_1 - p_0, n - p_1}, \end{aligned} \tag{29}$$

in the large sample limit. Of course, this result is exactly true in the ordinary linear model special case (i.e., for a Gaussian response and the identity link function).

Further, and as before in the ordinary linear model setup, AIC is obviously usable as an alternative model selection approach (in which case the models under comparison need not be nested).

The deviance can also provide alternative scaled residuals to the Pearson residuals. Indeed, and in practice, the distribution of the Pearson residuals can be quite asymmetric around zero, so that their behavior is not as close to ordinary linear model residuals as might be hoped for. The *deviance residuals* are often preferable in this respect. The deviance residuals are the generalization of the residuals $\hat{\epsilon}_i = y_i - \hat{\mu}_i$ from the ordinary linear model. They are constructed using the analogy that for ordinary linear models, as seen in (27), $D = \sum_{i=1}^n (y_i - \hat{\mu}_i)^2 = \sum_{i=1}^n \hat{\epsilon}_i^2$ is made up of the sum of the squared residuals. That is, the residuals are the square roots of the components of the deviance with the appropriate sign attached. So, writing d_i as the component of the deviance contributed by the ith observation (i.e., the ith term in the summation in (26)) we have

$$D = \sum_{i=1}^n d_i,$$

and, by analogy with the ordinary linear model, we can define

$$\hat{\epsilon}_i^D = \text{sign}(y_i - \hat{\mu}_i)\sqrt{d_i}.$$

As required, the sum of squares of these 'deviance residuals' gives the deviance itself. This definition has interesting distributional consequences. It can be shown that $\hat{\epsilon}_i^D$ follow, approximately, a $N(0, \phi)$ distribution if the model holds. The approximate normality in the deviance residuals allows to evaluate how well satisfied the assumption of the response distribution is. The speed of this asymptotic convergence and the effective validity of the approximation itself largely depends on several aspects: distribution of the response, sample size, and distribution of the predictors. It is however possible to have severe departures from normality even if the model is perfectly correct. See some examples here

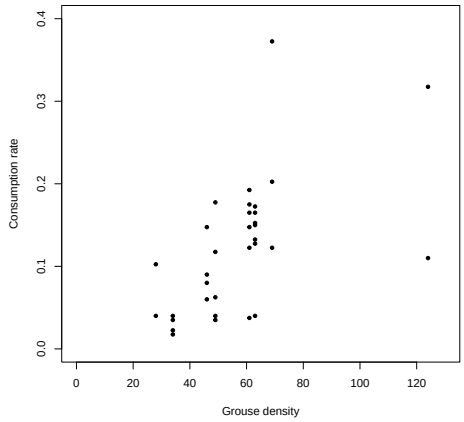
https://bookdown.org/egarpor/PM-UC3M/glm-diagnostics.html#fig:poisdiagnostics2-good

12 `glm` in R

The `glm` function provides the means for using GLMs in `R`. Its use is similar to that of the `lm` function but with two differences. The right hand side of the model formula, specifying the form for the linear predictor, now gives the link function of the mean of the response, rather than the mean of the response directly. Also `glm` takes a family argument, which is used to specify the distribution from the exponential family to use, and the link function that is to go with it.

To see the `glm` function in action, we consider the Hen Harrier data again. The data are available in the package `gamair`. First, let's read and plot the data.

```
library(gamair)
data(harrier)
#remove entries with zero consumption rate
harrier <- harrier[harrier$Consumption.Rate!=0, ]

plot(harrier$Grouse.Density, harrier$Consumption.Rate,
     ylim = c(0, 0.4), xlim = c(0, 130),
     xlab = "Grouse density", ylab = "Consumption rate", pch = 20)
```



Now fit the model discussed previously:

```
hm <- glm(Consumption.Rate ~ I(1/Grouse.Density^3),
          family = Gamma(link = "inverse"), data = harrier)
```

The argument `family` specifies the distribution (here `Gamma`) and the link function to use (here `"inverse"`). Let's now inspect the summary of the fitted model (stored in the object `hm`).

```
summary(hm)

##
## Call:
## glm(formula = Consumption.Rate ~ I(1/Grouse.Density^3), family = Gamma(link = "inverse"),
##     data = harrier)
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)           4.676e+00  1.156e+00   4.046 0.000321 ***
## I(1/Grouse.Density^3) 5.386e+05  1.655e+05   3.255 0.002741 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for Gamma family taken to be 0.347565)
##
##     Null deviance: 16.166  on 32  degrees of freedom
## Residual deviance: 10.472  on 31  degrees of freedom
## AIC: -92.383
##
## Number of Fisher Scoring iterations: 6
```

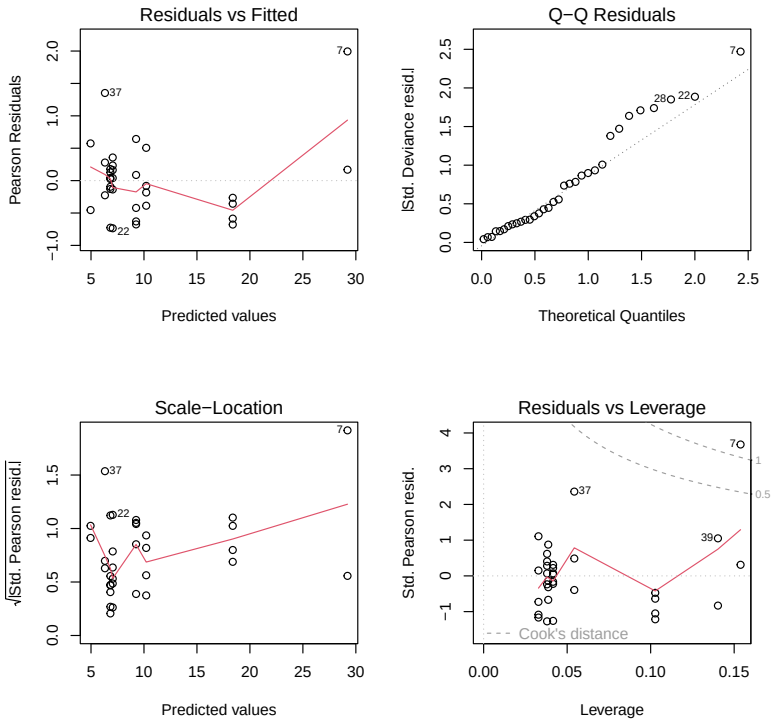
A few things to note from the summary of the fitted model. We know that for the Gamma distribution, the dispersion parameter is not one and so it has to be estimated, with one possibility (the only one covered in these notes) being (24). This is the estimator implemented by the `glm` function. Let's check this is indeed the case.

```
sum(residuals(hm, type = "pearson")^2)/(nrow(harrier) - 2)

## [1] 0.347565
```

If we want to obtain the deviance residuals, we just need to change the `type` argument in the function `residuals` from `pearson` to `deviance`. The residual deviance is the deviance of the model being fitted, while the null deviance is the deviance corresponding to a model with only an intercept term (which is the polar opposite of the saturated model).

As with any model, we should check residuals before proceeding.

```
par(mfrow = c(2, 2))
plot(hm)
```



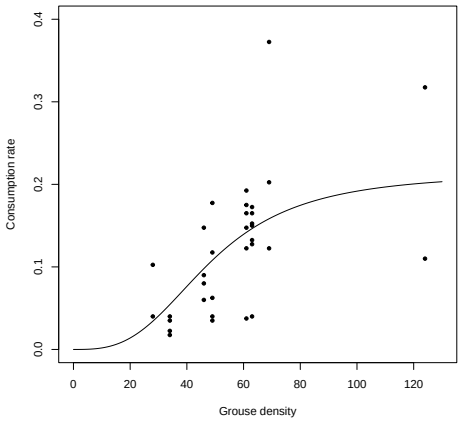
These residual plots are interpreted in much the same way as the equivalent plots for a linear model, with two differences: i) If the response distribution is not normal/Gaussian then we don't expect the normal QQ plot to show an exact straight line relationship and (ii) if our response is binary then checking is more difficult, with many plots being effectively useless.

For the harrier model there are some clear patterns in the residuals, with the data for some densities being entirely above or below the fitted line. For this simple example, it can help to plot model predictions and data on the same plot. To make predictions at a series of grouse densities, we can use the `predict` function.

```
pd <- data.frame(Grouse.Density = 0:130) #data at which to predict
fv <- predict(hm, pd, type = "response") #get model predictions
```

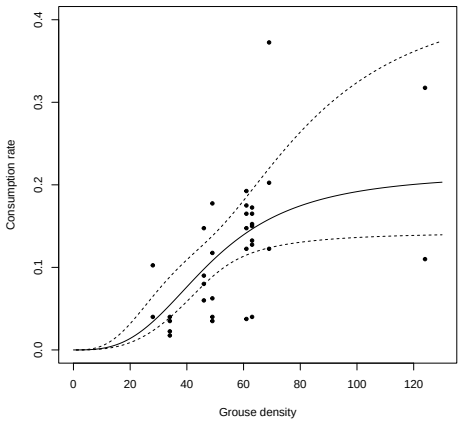
Note the `type` argument to `predict`, indicating that we want the predictions to be made on the original response scale, not the link transformed scale (the default). Now we can produce a plot.

```
plot(harrier$Grouse.Density, harrier$Consumption.Rate,
     ylim = c(0, 0.4), xlim = c(0, 130),
     xlab = "Grouse density", ylab = "Consumption rate", pch = 20)
lines(0:130, fv)
```



Actually it would be good to add approximate 95% CIs for the mean to the plot. Again `predict` is used, but this time standard errors for each prediction are also produced (on the link transformed scale, so that we need back-transformation afterwards to the original scale). Note that, unlike to what happens with an `lm` object, there is no `interval` argument for using `predict` with a `glm` object. The following adds CI's to the plot.

```
lp <- predict(hm, pd, se = TRUE)
plot(harrier$Grouse.Density, harrier$Consumption.Rate,
     ylim = c(0, 0.4), xlim = c(0, 130),
     xlab = "Grouse density", ylab = "Consumption rate", pch = 20)
lines(0:130, fv)
#phi was estimated so we need to use a t distribution
quant_t <- qt(0.975, df = (nrow(harrier) - 2))
lines(0:130, 1/(lp$fit + quant_t*lp$se.fit), lty = 2)
lines(0:130, 1/(lp$fit - quant_t*lp$se.fit), lty = 2)
```



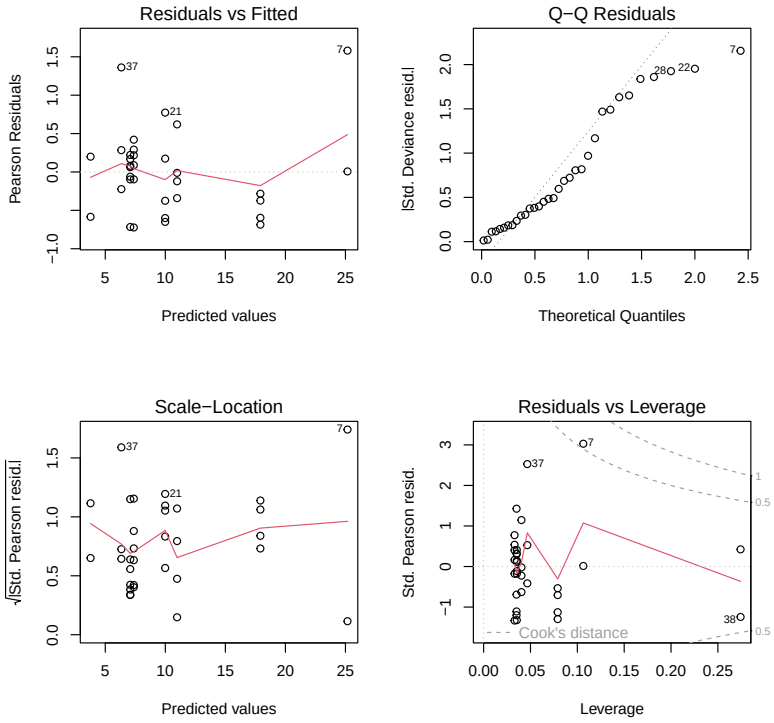
From this plot it is clear why we have patterns in the residuals, and only a very complex dependence of consumption rate on density would cure it. Since the different densities actually relate to different grouse moors, it is probable that there are some other moor-to-moor differences that should really be included in this model — perhaps relating to the alternative Hen-Harrier food available at each moor. Clearly some more work is required here.

Is the cubic dependence on density really the best model for these data? There are various ways of addressing this question, but one of the simplest is to try alternative powers, and compare the AICs of the alternative model fits. For example:

```
hm1 <- glm(Consumption.Rate ~ I(1/Grouse.Density),
           family = Gamma(link = "inverse"), data = harrier)

hm2 <- glm(Consumption.Rate ~ I(1/Grouse.Density^2),
           family = Gamma(link = "inverse"), data = harrier)
AIC(hm, hm1, hm2)

##     df       AIC
## hm   3 -92.38289
## hm1  3 -92.63887
## hm2  3 -94.17016
```

Out of curiosity, `df` is the number of parameters in each model (two regression coefficients and the dispersion parameter that in this case needs to be estimated). So AIC actually supports the model with a quadratic dependence on density. Note that this model also has 'better' diagnostic plots (although still with some slight issues).

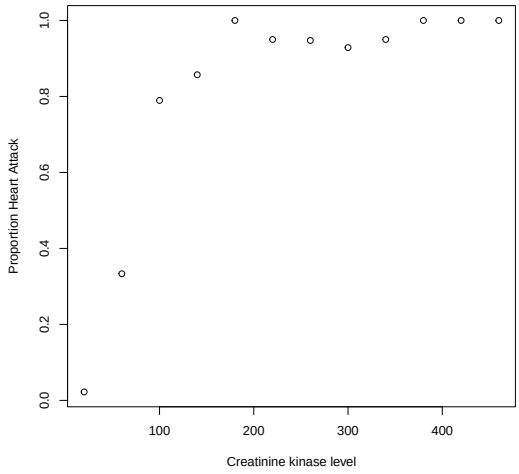
```
par(mfrow = c(2, 2))
plot(hm2)
```



12.1 Heart attack example

We revisit the heart attack data example from section 10.1. The goal is to construct a model which tries to explain the proportion of patients suffering a heart attack from the creatine kinase (levels). As usual, let's first read in the data and plot them.

```
ck<- c(20 , 60 , 100 , 140 , 180 , 220 , 260 , 300 , 340 , 380 , 420 , 460)
ha <- c(2 , 13 , 30 , 30 , 21 , 19 , 18 , 13 , 19 , 15 , 7 , 8 )
ok <- c( 88 , 26 , 8 , 5 , 0 , 1 , 1 , 1 , 1  , 0 , 0  , 0)
heart <- data.frame(ck = ck, ha = ha, ok = ok)

p <- heart$ha/(heart$ha + heart$ok)
plot(heart$ck, p, xlab = "Creatinine kinase level",
     ylab = "Proportion Heart Attack")
```



Recall that our basic model is

$$y_i \sim \text{Binomial}(N_i, p_i),$$

where y_i is the number of heart attack victims, observed out of N_i patients, with CK level x_i and p_i is the probability of heart attack at CK level x_i

$$p_i = \frac{e^{\beta_1+\beta_2 x_i}}{1+e^{\beta_1+\beta_2 x_i}}.$$

We have that $\mathbb{E}(y_i) \equiv \mu_i = N_i p_i$ and

$$g(\mu_i) = \log\left(\frac{\mu_i}{N_i-\mu_i}\right) = \log\left(\frac{p_i}{1-p_i}\right) = \text{logit}(p_i) = \eta_i = \beta_1 + \beta_2 x_i.$$

The logit link is the canonical link for binomial models, and hence the default in `R`. When using binomial models, we need to somehow supply the model fitting function with information about N_i as well as y_i. In `R` there are two ways of specifying binomial models with the `glm` function.

1. The response variable can be the observed proportion of successful binomial trials, in which case an array giving the number of trials must be supplied as the `weights` argument to `glm`. For binary data, no weights vector need be supplied, as the default weights of 1 suffice.

2. The response variable can be supplied as a two column array, in which the first column gives the number of binomial 'successes', and the second column is the number of binomial 'failures'.

Let's illustrate how exactly to do this.

```
#Approach 1
mod.0<- glm(heart$ha/(heart$ha + heart$ok) ~ ck,
            family = binomial(link = logit),
            weights = heart$ha + heart$ok,
            data = heart)
mod.0

##
## Call:  glm(formula = heart$ha/(heart$ha + heart$ok) ~ ck, family = binomial(link = logit),
##     data = heart, weights = heart$ha + heart$ok)
##
## Coefficients:
## (Intercept)           ck
##    -2.75836      0.03124
```

```
##
## Degrees of Freedom: 11 Total (i.e. Null);  10 Residual
## Null Deviance:     271.7
## Residual Deviance: 36.93  AIC: 62.33

#Approach 2
mod.0 <- glm(cbind(ha,ok) ~ ck,
             family = binomial(link = logit),
             data = heart)
mod.0

##
## Call:  glm(formula = cbind(ha, ok) ~ ck, family = binomial(link = logit),
##     data = heart)
##
## Coefficients:
## (Intercept)           ck
##    -2.75836      0.03124
##
## Degrees of Freedom: 11 Total (i.e. Null);  10 Residual
## Null Deviance:     271.7
## Residual Deviance: 36.93  AIC: 62.33
```

Note that supplying 2 arrays on the r.h.s. of the model formula involves using `cbind`. Note that we could have omitted the part `link = logit` since the logit link is the default for the binomial. As a matter of example, the null deviance (for a model with just an intercept) and the residual deviance (of the fitted model) can be combined to give the *proportion deviance explained*, a generalization of r^2, as follows:

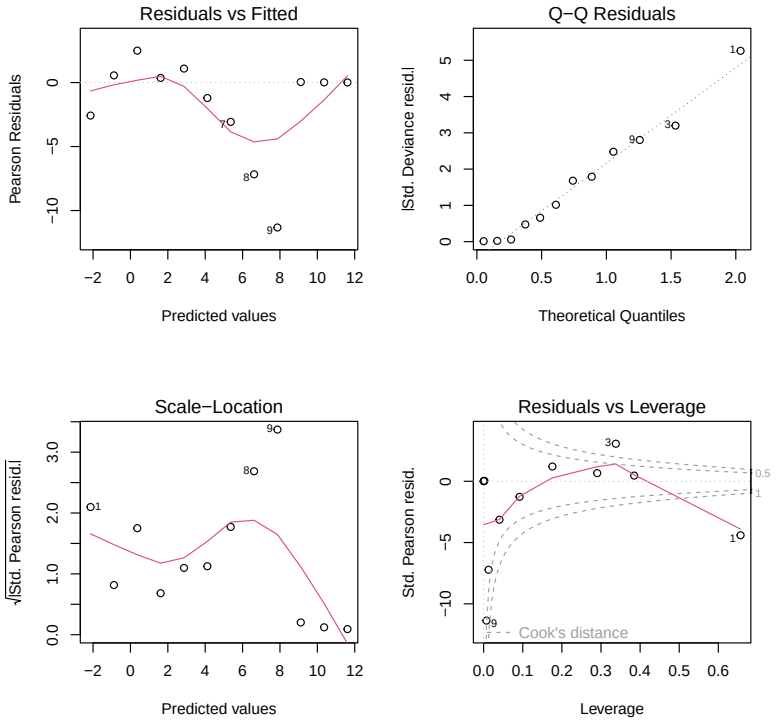
```
1 - (36.93/271.7)

## [1] 0.864078
```

Notice that the deviance is quite high for the χ^2_{10} random variable that it should approximate if the model is fitting well. In fact

```
pchisq(36.93, df = 10, lower.tail = FALSE)

## [1] 5.819325e-05
```

shows that there is a *very* small probability of a χ^2_{10} random variable being as large as 36.93. The residual plots also suggest a poor fit.

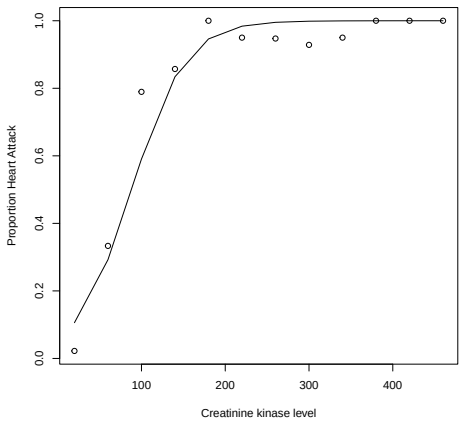
```
par(mfrow = c(2, 2))
plot(mod.0)
```



Again, the plots have much the same interpretation as the model checking plots for an ordinary linear model, except that it is now standardized residuals that are plotted, the `Predicted values` are on the scale of the linear predictor rather than the response, and some departure from a straight line relationship in the Normal QQ plot is often to be expected. The plots are not easy to interpret when there are so few data, but there appears to be a trend in the mean of the residuals plotted against predicted values, which would cause concern. Furthermore, the first point has very high influence.

Notice how the problems do not stand out so clearly from a plot of the fitted values overlayed on the raw estimated probabilities.

```
plot(heart$ck, p,
     xlab = "Creatinine kinase level",
     ylab = "Proportion Heart Attack")
lines(heart$ck, fitted(mod.0))
```

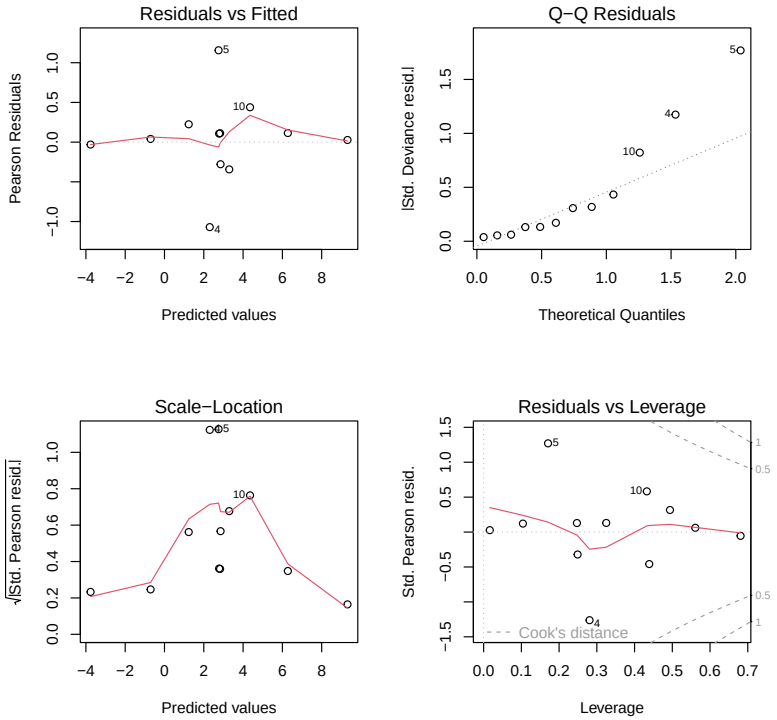


Note also that the fitted values provided by `glm` for binomial models are the estimated p_i's, rather than the estimated μ_i's ($= p_i N_i$).

The trend in the mean of the residuals suggests trying a cubic model, rather than the initial straight line:

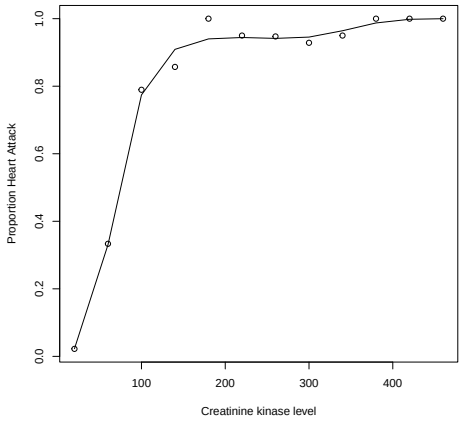
$$\text{logit}(p_i) = \beta_1 + \beta_2 x_i + \beta_3 x_i^2 + \beta_4 x_i^3.$$

```
mod.1 <- glm(cbind(ha,ok) ~ ck + I(ck^2) + I(ck^3),
             family = binomial, data = heart)
mod.1

##
## Call:  glm(formula = cbind(ha, ok) ~ ck + I(ck^2) + I(ck^3), family = binomial,
##     data = heart)
##
## Coefficients:
## (Intercept)           ck      I(ck^2)      I(ck^3)
##  -5.786e+00    1.102e-01   -4.649e-04    6.448e-07
##
## Degrees of Freedom: 11 Total (i.e. Null);  8 Residual
## Null Deviance:     271.7
## Residual Deviance: 4.252  AIC: 33.66

par(mfrow = c(2, 2))
plot(mod.1)
```



Clearly 4.252 is not too large for consistency with a χ^2_8 distribution (it is less than the expected value, in fact) and the AIC has improved substantially. The residual plots now show less clear patterns than for the previous model, although if we had more data then such a departure from constant variance would be a cause for concern.Furthermore the fit is clearly closer to the data now.

```
plot(heart$ck, p, xlab = "Creatinine kinase level",
     ylab = "Proportion Heart Attack")
lines(heart$ck, fitted(mod.1))
```



We can also get R to test the null hypothesis that `mod.0` is correct against the alternative that `mod.1` is required. Somewhat confusingly the `anova` function is used to do this, although it is a generalized likelihood ratio test that is being performed, and not an analysis of variance.

```
anova(mod.0, mod.1, test = "Chisq")

## Analysis of Deviance Table
##
## Model 1: cbind(ha, ok) ~ ck
## Model 2: cbind(ha, ok) ~ ck + I(ck^2) + I(ck^3)
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1        10     36.929
## 2         8      4.252  2   32.676 8.025e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

A p-value this low indicates very strong evidence against the null hypothesis: we are rejecting that β_3 and β_4 are both zero. Of course, we could still test whether we really need the cubic effect or if just the quadratic effect of CK would do it.

12.2 Melanoma case-control study

Case-control studies are an important type of study, in which a group of patients with some disease (the *cases*) are compared to a randomly selected group of healthy subjects from the same population as the cases (the *controls*). Variables that might be associated with the disease are also collected for all subjects. If a variable is really associated with the disease then it ought to be predictive of whether a randomly selected patient in the study is a case or a control. Such predictivity can be assessed using GLMs of the 'logistic regression type'.

For example, consider a study looking at 143 cases of malignant melanoma (a serious skin cancer) in white male patients aged 25 to 55, classified according to skin type (A, B, or C for 'celtic', 'middle european type', or ' Mediterranean'), compared to 356 white male controls aged 25-55 (selected without further reference to age or skin type). Patients were divided into 3 groups according to an 'age' factor variable, as well as being classified into 3 groups by the 'skin' factor variable (so there are 9 groups in total). The data are in a data frame called `md`:

```
md <- data.frame(mel = c(15, 8, 7, 26, 18, 6, 30, 25, 8),
                 stringsAsFactors = TRUE,
                 n = c(54, 52, 44, 75, 52, 42, 67, 66, 47),
```

```
                  age = rep(c("25-35", "35-45", "45-55"), each = 3),
                  skin = rep(c("A", "B", "C"), 3))
md
```

```
##   mel  n   age skin
## 1  15 54 25-35    A
## 2   8 52 25-35    B
## 3   7 44 25-35    C
## 4  26 75 35-45    A
## 5  18 52 35-45    B
## 6   6 42 35-45    C
## 7  30 67 45-55    A
## 8  25 66 45-55    B
## 9   8 47 45-55    C
```

In the `md` dataframe, the variable `mel` records the number of melanoma cases, while n is the total number of patients (melanoma cases + controls) for a given age band and skin type. Note that skin type is obviously a factor variable. The age variable, although its levels have an order, we will be considering it as a factor variable as well. Consider testing the null hypothesis that skin type is not associated with melanoma, against the alternative that it is. Neglecting the possibility of an interaction term, we could do this by comparing the null model

$$\text{logit}(p_i) = \mu + \beta_j, \quad y_i \sim \text{binom}(p_i, n_i),$$

to the alternative

$$\text{logit}(p_i) = \mu + \beta_j + \gamma_k, \quad y_i \sim \text{binom}(p_i, n_i), \tag{30}$$

where j is 1, 2, 3, corresponding to age-band level '25-35', '35-45', and '45-55', respectively, and $k \in \{1, 2, 3\}$ for skin type A, B, and C, respectively. Here, y_i is the number of melanoma cases, out of n_i subjects (cases + controls), in group i, for $i = 1, \ldots, 9$.

Fitting these models and comparing them using a generalized likelihood ratio test (aka analysis of deviance) is straightforward.

```
mel0 <- glm(mel/n ~ age, family = binomial, data = md, weights = n)
mel1 <- glm(mel/n ~ age + skin, family = binomial, data = md, weights = n)
anova(mel0, mel1, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: mel/n ~ age
## Model 2: mel/n ~ age + skin
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1         6    20.5062
## 2         4     3.4389  2   17.067 0.0001967 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note that `test = "Chisq"` specifies that a generalized likelihood ratio test is to be performed using (17). Here the p-value is very low: there is a very small probability of observing this large a difference in deviance between the two models if `mel0` really generated the data. This strongly suggests that `mel0` is incorrect. In other words, there is strong evidence in favour of `mel1` and an effect of skin type on melanoma risk. As an alternative to the `anova` function we could have used the `drop1` function, using also the argument `test = "Chisq"`.

```
drop1(mel1, test = "Chisq")
```

```
## Single term deletions
##
## Model:
## mel/n ~ age + skin
```

```
##        Df Deviance    AIC     LRT  Pr(>Chi)
## <none>      3.4389 50.569
## age     2  12.3876 55.517  8.9487 0.0113976 *
## skin    2  20.5062 63.636 17.0673 0.0001967 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The next step would be to examine the model coefficients to ascertain the nature and size of the effect. Recall that from the solutions of workshop 3 we already know how to interpret the coefficients of a logistic regression model when the predictor variable(s) is(are) continuous. Let's use this example to illustrate how to interpret the coefficients with factor predictors. We first note that we need to impose identiafiability constraints (as is always the case when factors are involved). We know that `R` by default sets the first level of each factor variable to the baseline (i.e., in practice this corresponds to removing the associated columns from the design matrix). The first levels in this case are skin type A and age band 25-35. Let's convince ourselves this is indeed the case.

```
levels(md$age)

## [1] "25-35" "35-45" "45-55"

levels(md$skin)

## [1] "A" "B" "C"
```

We could alternatively check the model matrix.

```
model.matrix(mel1)

##   (Intercept) age35-45 age45-55 skinB skinC
## 1           1        0        0     0     0
## 2           1        0        0     1     0
## 3           1        0        0     0     1
## 4           1        1        0     0     0
## 5           1        1        0     1     0
## 6           1        1        0     0     1
## 7           1        0        1     0     0
## 8           1        0        1     1     0
## 9           1        0        1     0     1
## attr(,"assign")
## [1] 0 1 1 2 2
## attr(,"contrasts")
## attr(,"contrasts")$age
## [1] "contr.treatment"
##
## attr(,"contrasts")$skin
## [1] "contr.treatment"
```

Obviously, this identifiability restrictions were already used by the `anova` and `drop1` functions. Indeed and, for example, in the output of the anova command we have that the residual degrees of freedom for the model only containing the effect of age is 6 which corresponds to the number of groups, 9, minus the number of parameters estimated in that model (3, the intercept and the parameters associated with two of the age band levels). Similarly, the residual degrees of freedom for the model containing the effects of both age and skin type is 4 which corresponds to 9 minus the 5 parameters that are being estimated in this model (the intercept, the parameters associated with two of the age band levels and the parameters associated with two of the skin type levels).

Note that we can rewrite the model in (30) using indicator/dummy variables as

$$\text{logit}(p_i) = \mu + \beta_1 \text{age}_{35-45,i} + \beta_2 \text{age}_{45-55,i} + \gamma_1 \text{skin}_{B,i} + \gamma_2 \text{skin}_{C,i}, \quad i = 1, \ldots, 9,$$

where

$$\text{age}_{35-45,i} = \begin{cases} 1, \text{ if group i is in age-band level 35-45,} \\ 0, \text{ otherwise,} \end{cases}$$

$$\text{age}_{45-55,i} = \begin{cases} 1, \text{ if group i is in age-band level 45-55,} \\ 0, \text{ otherwise,} \end{cases}$$

$$\text{skin}_{B,i} = \begin{cases} 1, \text{ if skin type is B for group i,} \\ 0, \text{ otherwise,} \end{cases}$$

$$\text{skin}_{C,i} = \begin{cases} 1, \text{ if skin type is C for group i,} \\ 0, \text{ otherwise.} \end{cases}$$

What is the interpretation of β_1? It is the difference in log odds of melanoma between the group whose age-band level is 35-45 and the reference age group (age band level 25-35), with both groups sharing the same skin type. To see why, and for the sake of concreteness, assume the skin type for both groups is B. We are then effectively comparing groups 2 and 5 in the dataframe order. We have that:

$$\text{logit}(p_5) = \log\left(\frac{p_5}{1-p_5}\right) = \mu + \beta_1 \times 1 + \beta_2 \times 0 + \gamma_1 \times 1 + \gamma_2 \times 0 = \mu + \beta_1 + \gamma_1,$$

$$\text{logit}(p_2) = \log\left(\frac{p_2}{1-p_2}\right) = \mu + \beta_1 \times 0 + \beta_2 \times 0 + \gamma_1 \times 1 + \gamma_2 \times 0 = \mu + \gamma_1.$$

Therefore,

$$\log\left(\frac{p_5}{1-p_5}\right) - \log\left(\frac{p_2}{1-p_2}\right) = (\mu + \beta_1 + \gamma_1) - (\mu + \gamma_1) = \beta_1.$$

It is easy to see that if replace skin type B by A or C, we would still obtain β_1 (the crux is that the two groups being compared share the same skin type and there is no interaction between age and skin type). We further have

$$\log\left(\frac{p_5}{1-p_5}\right) - \log\left(\frac{p_2}{1-p_2}\right) = \beta_1 \Rightarrow \log\left(\frac{\frac{p_5}{1-p_5}}{\frac{p_2}{1-p_2}}\right) = \beta_1 \Rightarrow \frac{p_5}{1-p_5} = \frac{p_2}{1-p_2} e^{\beta_1}.$$

That is, for two groups with the same skin type, the odds of melanoma for those in age band 35-45 is e^{β_1} times the odds of those in the reference group of age (age band 25-35). Similarly, for two groups with the same skin type, the odds of melanoma for those in age band 45-55 is e^{β_2} times the odds of those in the reference group of age (age band 25-35). Now, and applying the same reasoning, for two groups in the same age-band, the odds of melanoma for those with skin type B is e^{γ_1} times the odds of those in the reference group of skin type (i.e., skin type A). Finally, for two groups in the same age-band, the odds of melanoma for those with skin type C is e^{γ_2} times the odds of those in the reference group of skin type (i.e., skin type A). Let's look at the coefficients.

```
mel1
```

```
##
## Call:  glm(formula = mel/n ~ age + skin, family = binomial, data = md,
##     weights = n)
##
## Coefficients:
## (Intercept)     age35-45     age45-55        skinB        skinC
##     -1.0228       0.4766       0.7655      -0.2881      -1.1063
##
## Degrees of Freedom: 8 Total (i.e. Null);  4 Residual
## Null Deviance:     29.89
## Residual Deviance: 3.439  AIC: 50.57
```

```
exp(coefficients(mel1))
```

```
## (Intercept)    age35-45    age45-55       skinB       skinC
##   0.3595763   1.6106085   2.1499647   0.7496932   0.3307686
```

We conclude that for two groups with the same skin type, the estimated odds of melanoma for those in age band 35-45 is 1.61 times the odds of those in the reference group of age (age band 25-35). Therefore, there is an increase in the odds of disease when age band moves from baseline to the 'next level'. The estimated increase in odds is even bigger when comparing the baseline age band and age band 45-55, as $e^{\beta_2} = 2.15$ (keeping, of course, skin type the same for the two groups). On the other hand, for two groups in the same age-band, the estimated odds of melanoma for those with skin type B is 0.75 times the odds of those with skin type A (so an estimated decrease in the odds of disease). Finally, for two groups in the same age-band, the estimated odds of melanoma for those with skin type C is 0.33 times the odds of those with skin type A (an even more accentuated decrease in the odds of melanoma).

Of course, we should attach confidence intervals to this statements and we are already in position of doing this. We know from (22) that the maximum likelihood estimates of the parameters follow, in the large sample limit, a (multivariate) normal distribution with mean vector equal to the true (but, of course, unknown) parameters and with covariance matrix given by $\phi(\mathbf{X}^\mathsf{T}\mathbf{W}\mathbf{X})^{-1}$. Because for the binomial distribution, $\phi = 1$, this result can be used directly to find confidence intervals for the parameters. For instance, we know that a $100(1-\alpha)\%$ confidence interval for β_1 is given by

$$(\hat{\beta}_1 - z_{1-\alpha/2}\sigma_{\hat{\beta}_1}, \hat{\beta}_1 + z_{1-\alpha/2}\sigma_{\hat{\beta}_1}),$$

where $\sigma^2_{\hat{\beta}_1}$ is one of the diagonal entries of the covariance matrix $(\mathbf{X}^\mathsf{T}\mathbf{W}\mathbf{X})^{-1}$. We then trivially have that a $100(1-\alpha)\%$ confidence interval for e^{β_1} is

$$(e^{\hat{\beta}_1 - z_{1-\alpha/2}\sigma_{\hat{\beta}_1}}, e^{\hat{\beta}_1 + z_{1-\alpha/2}\sigma_{\hat{\beta}_1}}).$$

All this can be easily obtained from the summary output.

```
summary(mel1)

##
## Call:
## glm(formula = mel/n ~ age + skin, family = binomial, data = md,
##     weights = n)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.0228     0.2355  -4.343 1.41e-05 ***
## age35-45      0.4766     0.2689   1.773  0.07630 .
## age45-55      0.7655     0.2611   2.932  0.00337 **
## skinB        -0.2881     0.2267  -1.271  0.20381
## skinC        -1.1063     0.2829  -3.911 9.19e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 29.8893  on 8  degrees of freedom
## Residual deviance:  3.4389  on 4  degrees of freedom
## AIC: 50.569
##
## Number of Fisher Scoring iterations: 4
```

We can thus easily compute a 95% CI for e^{β_1}.

```
lower_bound <- exp(0.4766 - qnorm(0.975)*0.2689)
upper_bound <- exp(0.4766 + qnorm(0.975)*0.2689)
lower_bound; upper_bound

## [1] 0.9508164
## [1] 2.728179
```

Note how the confidence interval contains the value $1 = e^0$. Confidence intervals for the other parameters are obtained in a similar fashion. We can, in fact, compute all intervals in an automatic way through the use of the `confint.default` function.

```
exp(confint.default(mel1))
```

```
##                 2.5 %    97.5 %
## (Intercept) 0.2266304 0.5705107
## age35-45    0.9508683 2.7280957
## age45-55    1.2887860 3.5865911
## skinB       0.4807427 1.1691073
## skinC       0.1899965 0.5758417
```

Finally, these case-control studies allow estimation of all regression coefficients with the exception of the intercept (which in this case corresponds to the log odds of melanoma for group 1, the baseline group, i.e., the group with age band 25-35 and skin type A) because we have chosen the ratio of cases to controls, rather than observing it in the population of interest. Therefore μ does not provide an estimate of the population log odds of disease (melanoma) for the baseline group. As a consequence, we cannot estimated the probabilities of disease for each group

$$p_i = \frac{e^{\mu+\beta_1 \text{age}_{35-45,i}+\beta_2 \text{age}_{45-55,i}+\gamma_1 \text{skin}_{B,i}+\gamma_2 \text{skin}_{C,i}}}{1+e^{\mu+\beta_1 \text{age}_{35-45,i}+\beta_2 \text{age}_{45-55,i}+\gamma_1 \text{skin}_{B,i}+\gamma_2 \text{skin}_{C,i}}}, \quad i = 1, \ldots, 9,$$

as they involve μ. However, and fortunately, estimates of the standard deviations of slope coefficients and hypothesis testing techniques like the generalized likelihood ratio test can still be used with case-control data. Note that although `R` returns an estimate of μ we should not use it. It is just that `R` does not have a way of knowing that data are from a case-control study.

12.3 Insurance claims testing example

The dataset `motori` is derived from dataset `motorins` from R library `faraway`. It contains insurance company data from Sweden, on payouts (`Payment`) in relation to number `Insured`, `km` travelled (a numeric variable with 5 discrete values), `Make` of car (a factor variable with 9 levels representing different car models), and number of years with no claims `Bonus`. For more information please type `motorins`. An initial model for the data is:

$$\mathbb{E}(\texttt{Payment}_i) = \texttt{Insured}_i \times \text{risk}_i, \quad \text{where } \texttt{Payment} \sim \text{gamma},$$

so

$$\log\{\mathbb{E}(\texttt{Payment}_i)\} = \log(\texttt{Insured}_i) + \log(\text{risk}_i).$$

$\log(\texttt{Insured}_i)$ is an example of a model *offset* — a predictor variable whose coefficient is fixed at 1. This is included because one expects that the total amount of claims, and hence the payment amount, for a group will, in general, be proportionate to the number of insured. $\log(\text{risk}_i)$ can be modelled using a linear model structure to give:

$$\log\{\mathbb{E}(\texttt{Payment}_i)\} = \log(\texttt{Insured}_i) + \alpha + \beta_j + \gamma \texttt{km}_i + \delta_j \texttt{km}_i + \zeta \texttt{Bonus}_i,$$

where β_j is the main effect for `Make`, $j \in \{1, \ldots, 9\}$, $\gamma \texttt{km}_i$ is the main effect of kms traveled, $\delta_j \texttt{km}_i$ is the `Make`-km interaction, and $\zeta \texttt{Bonus}_i$ is the effect of Bonus. Of course this model is not identifiable and one possibility to identify it is to set the first level of the `Make` variable to zero (the default in `R`). Consider testing the null hypothesis that the effect of `km`s travelled on payment does not vary with the car model (encoded in the `Make` variable), i.e., consider testing the null hypothesis that the interaction term between the `Make` variable and `km`s is not needed in the model. First, let's fit the corresponding null and full models, i.e., the models without and with the interaction term, respectively.

```
require(faraway)
data(motorins)

#attention will be restricted to data from zone 1
```

```
motori <- motorins[motorins$Zone==1,]
motori$km <- as.numeric(motori$Kilometres)
#we are not using the variables perd and claims
head(motori)

##   Kilometres Zone Bonus Make Insured Claims Payment     perd km
## 1          1    1     1    1  455.13    108  392491 3634.176  1
## 2          1    1     1    2   69.17     19   46221 2432.684  1
## 3          1    1     1    3   72.88     13   15694 1207.231  1
## 4          1    1     1    4 1292.39    124  422201 3404.847  1
## 5          1    1     1    5  191.01     40  119373 2984.325  1
## 6          1    1     1    6  477.66     57  170913 2998.474  1

#full model
g1 <- glm(Payment ~ offset(log(Insured)) + km*Make + Bonus,
          family = Gamma(link = log), data = motori)
#null model
g0 <- glm(Payment ~ offset(log(Insured)) + km + Make + Bonus,
          family = Gamma(link = log), data = motori)
```

Since ϕ is not known for the gamma distribution, we now perform an F-ratio test comparing the models using (29).

```
anova(g0, g1, test = "F")

## Analysis of Deviance Table
##
## Model 1: Payment ~ offset(log(Insured)) + km + Make + Bonus
## Model 2: Payment ~ offset(log(Insured)) + km * Make + Bonus
##   Resid. Df Resid. Dev Df Deviance     F Pr(>F)
## 1       284     155.06
## 2       276     151.89  8    3.166 0.752 0.6455
```

It appears that the dependence of the payouts on km travelled can be assumed not to vary with car make. The residual plots, both for model `g0` and `g1`, show that the mean variance relationship implied by the Gamma distribution is not appropriate for these data and hence all conclusions need to be tempered. Let us look at the summary of model `g0` just for the sake of illustrating how to interpret the estimated coefficients. The final model is

$$\log\{\mathbb{E}(\texttt{Payment}_i)\} = \log(\texttt{Insured}_i) + \alpha + \beta_j + \gamma\texttt{km}_i + \zeta\texttt{Bonus}_i,$$

or, equivalently,

$$\mathbb{E}(\texttt{Payment}_i) = \exp\{\log(\texttt{Insured}_i) + \alpha + \beta_j + \gamma\texttt{km}_i + \zeta\texttt{Bonus}_i\}.$$

```
summary(g0)

##
## Call:
## glm(formula = Payment ~ offset(log(Insured)) + km + Make + Bonus,
##     family = Gamma(link = log), data = motori)
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.52731    0.17774  36.723  < 2e-16 ***
## km           0.12007    0.03115   3.855 0.000143 ***
## Make2        0.40704    0.17824   2.284 0.023128 *
## Make3        0.15535    0.17955   0.865 0.387666
```

```
## Make4       -0.34394    0.19150  -1.796 0.073546 .
## Make5        0.14467    0.18097   0.799 0.424726
## Make6       -0.34557    0.17824  -1.939 0.053515 .
## Make7        0.06136    0.18244   0.336 0.736893
## Make8        0.75038    0.18726   4.007 7.85e-05 ***
## Make9        0.03197    0.17824   0.179 0.857783
## Bonus       -0.20069    0.02150  -9.334  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 0.5559565)
##
##     Null deviance: 238.97  on 294  degrees of freedom
## Residual deviance: 155.06  on 284  degrees of freedom
## AIC: 7167.5
##
## Number of Fisher Scoring iterations: 6
```

The Bonus variable is a discrete (in practical terms interpreted as continuous) one and its interpretation is as follows: the expected payout changes by a factor of $e^{-0.20069} = 0.82$ for every increasing year since last claim (when keeping, of course, kms travelled and car model variables values fixed and also the number of insured). On the other hand, the `Make` variable is a factor representing different car models (levels range from 1 to 9 and baseline is level 1) and therefore, the expected payout for car model category, say 2, is $e^{0.40704} = 1.5$ times that of category 1 (when keeping kms travelled, bonus, and number of insured values fixed).

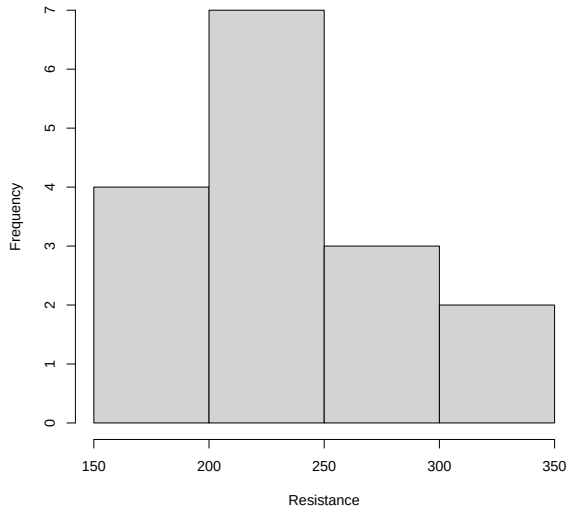
12.4 Semiconductor wafer model selection example

As an example of backwards selection in a GLM context, we consider the semiconductor electrical resistance data available in the data frame `wafer` in the package `faraway`. Four factors (`x1` to `x4`), with levels `+` and `-`, in the manufacturing process were believed to influence semiconductor resistance, `resist`, and an experiment was conducted to try out all combinations of two levels of each. Previous analyses of this dataset, to take into account the skewness in the response variable (see below) used an ordinary linear model with a log transformation of the response. Here we will use a Gamma distribution (which is positive and skewed) with a log link. The data are as follows.

```
require(faraway)
wafer
```

```
##    x1 x2 x3 x4 resist
## 1   -  -  -  -  193.4
## 2   +  -  -  -  247.6
## 3   -  +  -  -  168.2
## 4   +  +  -  -  205.0
## 5   -  -  +  -  303.4
## 6   +  -  +  -  339.9
## 7   -  +  +  -  226.3
## 8   +  +  +  -  208.3
## 9   -  -  -  +  220.0
## 10  +  -  -  +  256.4
## 11  -  +  -  +  165.7
## 12  +  +  -  +  203.5
## 13  -  -  +  +  285.0
## 14  +  -  +  +  268.0
## 15  -  +  +  +  169.1
## 16  +  +  +  +  208.5
```

```
hist(wafer$resist, xlab = "Resistance", main = "")
```



An initial model is fitted with all interactions of the factors up to 3rd order. Note that this leads to a model with 15 parameters and we only have 16 observations (almost the saturated model!) so some model selection is really needed here.

```
wm <- glm(resist ~ (x1 + x2 + x3 + x4)^3,
          Gamma(link = "log"), data = wafer)
```

Applying backward selection, we now want to calculate the p-values associated with dropping each 3 way interaction from the full model (one at a time!). Recall that the `drop1` function in R is a very convenient way of doing this. It goes through all the terms in the model that can be dropped on their own, refits without each of them in turn and compares each resulting fit with the original model. Remember that the dispersion parameter ϕ needs to be estimated for the Gamma distribution and so we need to conduct a F-ratio test.

```
drop1(wm, test = "F")

## Single term deletions
##
## Model:
## resist ~ (x1 + x2 + x3 + x4)^3
##          Df  Deviance    AIC F value Pr(>F)
## <none>       0.0083651 129.73
## x1:x2:x3  1 0.0086826 127.76  0.0380 0.8775
## x1:x2:x4  1 0.0285857 130.14  2.4173 0.3639
## x1:x3:x4  1 0.0109530 128.03  0.3094 0.6769
## x2:x3:x4  1 0.0107629 128.01  0.2867 0.6871
```

The rows of the table are labelled with the names of the dropped terms. The reported p-value in the final row is for testing the null hypothesis that the model without the dropped term is adequate (against the alternative that the full model is needed). The AIC for each model under consideration is also reported. The p-values suggest dropping the interaction `x1:x2:x3`, and

```
wm1 <- update(wm, .~. - x1:x2:x3)
drop1(wm1, test = "F")

## Single term deletions
##
## Model:
```

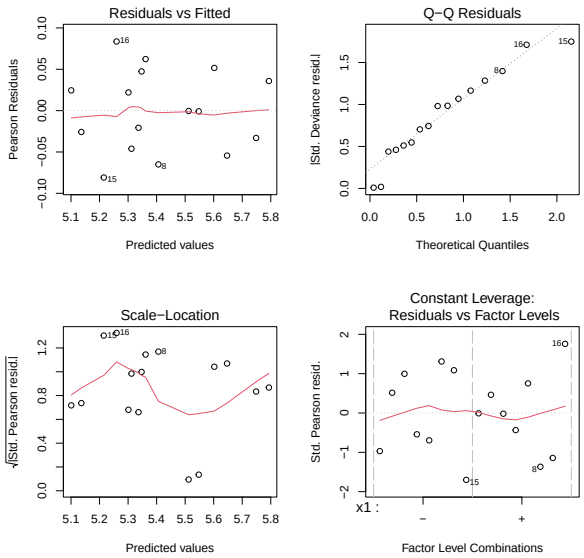
```
## resist ~ x1 + x2 + x3 + x4 + x1:x2 + x1:x3 + x1:x4 + x2:x3 +
##     x2:x4 + x3:x4 + x1:x2:x4 + x1:x3:x4 + x2:x3:x4
##          Df  Deviance    AIC F value Pr(>F)
## <none>      0.0086826 128.32
## x1:x2:x4  1 0.0289028 130.98  4.6577 0.1636
## x1:x3:x4  1 0.0112704 126.92  0.5961 0.5208
## x2:x3:x4  1 0.0110804 126.87  0.5523 0.5348
```

Notice how they have changed from the previous set of p-values. Now the `x3:x4:x2` interaction is the one to drop. Repeating these steps we eventually end up with

```
wm7 <-glm(resist ~ x1*x3 + x3*x4 + x2*x3,
          Gamma(link = "log"), data = wafer)
drop1(wm7, test = "F")

## Single term deletions
##
## Model:
## resist ~ x1 * x3 + x3 * x4 + x2 * x3
##        Df Deviance    AIC F value  Pr(>F)
## <none>    0.036266 139.20
## x1:x3   1 0.060433 142.54  5.3311 0.04977 *
## x3:x4   1 0.069345 144.51  7.2970 0.02702 *
## x3:x2   1 0.067314 144.06  6.8491 0.03079 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

summary(wm7)

##
## Call:
## glm(formula = resist ~ x1 * x3 + x3 * x4 + x2 * x3, family = Gamma(link = "log"),
##     data = wafer)
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.31195    0.04757 111.677 4.62e-14 ***
## x1+          0.20029    0.04757   4.211  0.00295 **
## x3+          0.43674    0.06727   6.493  0.00019 ***
## x4+          0.03537    0.04757   0.744  0.47836
## x2+         -0.21101    0.04757  -4.436  0.00218 **
## x1+:x3+     -0.15549    0.06727  -2.312  0.04957 *
## x3+:x4+     -0.18195    0.06727  -2.705  0.02687 *
## x3+:x2+     -0.17626    0.06727  -2.620  0.03064 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 0.004524942)
##
##     Null deviance: 0.697837  on 15  degrees of freedom
## Residual deviance: 0.036266  on  8  degrees of freedom
## AIC: 139.2
##
## Number of Fisher Scoring iterations: 4
```

So if 0.05 is the threshold for retaining a term then this is the final model. In this model all main effects are present, along with 3 two way interactions, all involving `x3`. The residual plots are ok, although sample size is admittedly very small here.

```
par(mfrow = c(2, 2))
plot(wm7)
```



Once a model has been selected, then the coefficient estimates would be examined and interpreted, possibly with the aid of confidence intervals. Note however that inference performed with the final fitted model tends to overstate what can really be concluded from the data, since it does not allow for the uncertainty in model selection. To finalize, let's write down mathematically the selected model.

$$
\begin{aligned}
y_i = \text{resist}_i \sim \text{gamma}, \quad \mathbb{E}(y_i) = \mu_i, \quad i = 1, \ldots, 16, \\
g(\mu_i) = \log(\mu_i) = \beta_0 + \beta_1 x_{1,i}^{+} + \beta_2 x_{2,i}^{+} + \beta_3 x_{3,i}^{+} + \beta_4 x_{4,i}^{+} + \beta_5 x_{1,i}^{+} x_{3,i}^{+} + \beta_6 x_{2,i}^{+} x_{3,i}^{+} + \beta_7 x_{3,i}^{+} x_{4,i}^{+}, \qquad (31)
\end{aligned}
$$

where $x_{1,i}^{+}$ is a dummy variable taking the value 1 if `x1` takes the value + and 0 otherwise. The other variables, $x_{2,i}^{+}$, $x_{3,i}^{+}$, and $x_{4,i}^{+}$ are similarly defined. The estimates of each parameter $(\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_7)$ are obtained from the summary output above. As before, and if it helps, we can also look at the model matrix (note that we have written the model in (31) using the identifiability constraints imposed in `R` by default, so that the information we get from the model and the `model.matrix` is 'aligned').

```
model.matrix(wm7)
```

```
##    (Intercept) x1+ x3+ x4+ x2+ x1+:x3+ x3+:x4+ x3+:x2+
## 1            1   0   0   0   0       0       0       0
## 2            1   1   0   0   0       0       0       0
## 3            1   0   0   0   1       0       0       0
## 4            1   1   0   0   1       0       0       0
## 5            1   0   1   0   0       0       0       0
## 6            1   1   1   0   0       1       0       0
## 7            1   0   1   0   1       0       0       1
## 8            1   1   1   0   1       1       0       1
## 9            1   0   0   1   0       0       0       0
## 10           1   1   0   1   0       0       0       0
## 11           1   0   0   1   1       0       0       0
## 12           1   1   0   1   1       0       0       0
## 13           1   0   1   1   0       0       1       0
## 14           1   1   1   1   0       1       1       0
## 15           1   0   1   1   1       0       1       1
## 16           1   1   1   1   1       1       1       1
```

```
## attr(,"assign")
## [1] 0 1 2 3 4 5 6 7
## attr(,"contrasts")
## attr(,"contrasts")$x1
## [1] "contr.treatment"
##
## attr(,"contrasts")$x3
## [1] "contr.treatment"
##
## attr(,"contrasts")$x4
## [1] "contr.treatment"
##
## attr(,"contrasts")$x2
## [1] "contr.treatment"
```

The model matrix basically tell us the circumstance under which the coefficient of a factor will be added to the model. The `intercept` (denoted by β_0 in model (31)) is the expected resistance for a wafer where none of the factors are in the + state. The coefficients associated to `x1+` to `x4+` (β_1 to β_4 in (31)) give the expected increase in resistivity when just one of the factors is in the + state (referring back to the summary, `x1` and `x3` seem to lead to a significant increase in resistance, on their own). So what about the interactions? Look at `x1+:x3+` as an example — it's an adjustment that is added on when `x1` and `x3` are *both* in the + state together: i.e. it is how much different the expected resistivity is to what you would expect if the effects of `x1` and `x3` both being + simply added to each other. Referring back to the summary, it seems that when two factors are in the + state, the resistivity is lower than you would expect from just looking at their effects on their own.

12.4.1 AIC based selection

As an example of using AIC let's redo the wafer model selection example using AIC for backwards selection. This is conceptually no different to what we have done before with ordinary linear models. Since AIC is what `drop1` does by default, we do not have to specify any further argument, only pass our `wm` fitted object (which contains all interactions of the four factors up to the third order).

```
drop1(wm)

## Single term deletions
##
## Model:
## resist ~ (x1 + x2 + x3 + x4)^3
##          Df  Deviance    AIC
## <none>      0.0083651 129.73
## x1:x2:x3  1 0.0086826 127.76
## x1:x2:x4  1 0.0285857 130.14
## x1:x3:x4  1 0.0109530 128.03
## x2:x3:x4  1 0.0107629 128.01
```

The model with the lowest AIC should be selected. In this instance it is the model that omits `x1:x2:x3`, so that term will be dropped.

```
wm1 <- update(wm, .~. - x1:x2:x3)
drop1(wm1)

## Single term deletions
##
## Model:
## resist ~ x1 + x2 + x3 + x4 + x1:x2 + x1:x3 + x1:x4 + x2:x3 +
##     x2:x4 + x3:x4 + x1:x2:x4 + x1:x3:x4 + x2:x3:x4
##          Df  Deviance    AIC
```

```
## <none>      0.0086826 128.32
## x1:x2:x4  1 0.0289028 130.98
## x1:x3:x4  1 0.0112704 126.92
## x2:x3:x4  1 0.0110804 126.87
```

Notice one wrinkle: the AIC reported for `wm1` is 128.32 (remember that `<none>` corresponds to the model being fitted), but when we used `drop1` before on `wm`, it suggested that the AIC for `wm1` would be 127.76. This happens because we need a scale parameter (ϕ) estimate in order to evaluate the AIC, and `drop1` always uses the same estimate for all the models it compares, based on the largest model it is considering. Hence the two calls to `drop1` give different AIC estimates for the same model, because the different calls are using different scale parameter estimates. Of course the problem does not arise if ϕ is known.

As we already know, we can make the process fully automatic through the use of the `step` function, which can perform backward (the default), forward, or a hybrid strategy, based on the AIC. Let's do it!

```
step(wm)
```

```
## Start:  AIC=129.73
## resist ~ (x1 + x2 + x3 + x4)^3
##
##            Df  Deviance    AIC
## - x1:x2:x3  1 0.0086826 127.76
## - x2:x3:x4  1 0.0107629 128.01
## - x1:x3:x4  1 0.0109530 128.03
## <none>        0.0083651 129.73
## - x1:x2:x4  1 0.0285857 130.14
##
## Step:  AIC=128.32
## resist ~ x1 + x2 + x3 + x4 + x1:x2 + x1:x3 + x1:x4 + x2:x3 +
##     x2:x4 + x3:x4 + x1:x2:x4 + x1:x3:x4 + x2:x3:x4
##
##            Df  Deviance    AIC
## - x2:x3:x4  1 0.0110804 126.87
## - x1:x3:x4  1 0.0112704 126.92
## <none>        0.0086826 128.32
## - x1:x2:x4  1 0.0289028 130.98
##
## Step:  AIC=130.22
## resist ~ x1 + x2 + x3 + x4 + x1:x2 + x1:x3 + x1:x4 + x2:x3 +
##     x2:x4 + x3:x4 + x1:x2:x4 + x1:x3:x4
##
## Call:  glm(formula = resist ~ x1 + x2 + x3 + x4 + x1:x2 + x1:x3 + x1:x4 +
##     x2:x3 + x2:x4 + x3:x4 + x1:x2:x4 + x1:x3:x4, family = Gamma(link = "log"),
##     data = wafer)
##
## Coefficients:
## (Intercept)          x1+          x2+          x3+          x4+      x1+:x2+
##      5.2586       0.2843      -0.1273       0.4626       0.1502      -0.1228
##     x1+:x3+      x1+:x4+      x2+:x3+      x2+:x4+      x3+:x4+  x1+:x2+:x4+
##     -0.2071      -0.1852      -0.1783      -0.1863      -0.2339       0.2845
## x1+:x3+:x4+
##      0.1018
##
## Degrees of Freedom: 15 Total (i.e. Null);  3 Residual
## Null Deviance:     0.6978
## Residual Deviance: 0.01108  AIC: 130.2
```

Notice that the finally selected model is a little different to the one that was selected using hypothesis testing; this is no surprise: AIC tends to select more complex models (due to the nature of the criterion).

13 Mixed effects models (non-examinable)

Statistical models describe how data were sampled from a population about which we want to learn. To this point, our statistical models involved only *fixed* effects, that is, for all of our models up to now, the levels of the predictors used in the study were the only levels of interest to the analysis. As a result, we were restricted to make statistical inferences only over these specific levels. The key notion underlying fixed effects is that the levels used in the study are not randomly selected (from a larger population of possible levels). Almost always, quantitative/continuous predictors are fixed effects. Factor predictors are fixed effects when the levels are the only ones available and the goal of the analysis is to make inferences over these specific levels. In contrast, *random effects* represent a random sample from a much larger population of possible levels. Consider an experiment to investigate the effect of several drug treatments on a sample of patients. Typically, we are interested in specific drug treatments and so we would treat the drug effects as fixed. However, it makes most sense to treat the patient effects as random. The analyst selects the patients for the study from a large population of possible people. The focus of all statistical inference is not on the specific patients selected; rather, the focus is on the population of all possible patients. The key point underlying all random effects is this focus on the population and not on the specific units selected for the study. Random effects are almost always categorical.

Many studies involve mixed effects models where some predictors are fixed effects and some are random effects. Mixed effects models are useful for a wide array of study types. For instance, in medical studies, a subject's response to some treatment is monitored over time (a longitudinal study). Subjects in these studies are selected at random from a population of possible subjects. In wildlife studies, the abundance of some species is determined for each of several observation stations (transects) over several years. Observation stations such as transects are selected at random from an entire study area. The common thread in these these applications, and more general in applications involving random effects, is that the data are grouped into clusters according to the levels of one or more classification factors (by subjects and transects, respectively). Unlike the ordinary linear regression model and generalized linear models, mixed effects models are built to accommodate the inherent correlation that exists among observations within the same cluster. These models also enable the user to consider the clusters of observations (subjects, transects, etc.) as random samples from a common probability distribution, thus enabling the user to make more general interpretations.

13.1 A simple random effects model

The `Rail` data available in the `nlme` package are from an experiment in nondestructive testing for longitudinal stress in railway rails. Six rails were chosen at random and tested three times each by measuring the time it took for a certain type of ultrasonic wave to travel the length of the rail. The only experimental setting that changes between the observations is the rail.

```
library(nlme)
head(Rail)
```

```
## Grouped Data: travel ~ 1 | Rail
##   Rail travel
## 1    1     55
## 2    1     53
## 3    1     54
## 4    2     26
## 5    2     37
## 6    2     32
```

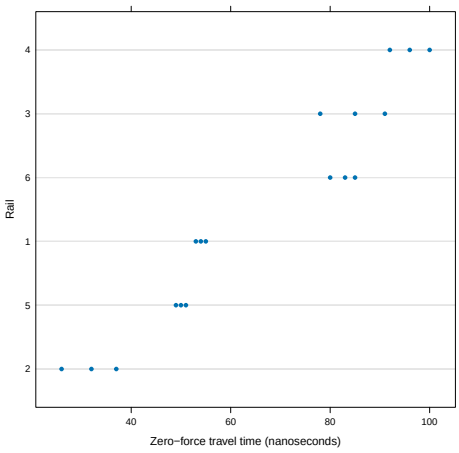
The structure of the data is quite simple: each row corresponds to one observation for which the rail and the travel time are recorded. The quantities of interest to the engineers, which they aimed to estimate from this experiment, are:

- The average travel time for a 'typical' rail (the expected travel time).
- The variation in average travel times among rails (the between-rail variability).

- The variation in the observed travel times for a single rail (the within-rail variability).

We can see in the figure below that there is considerable variability in the mean travel time for the different rails. Overall the between-rail variability is much greater than the within-rail variability.

```
plot(Rail)
```



The following simple random effects model is appropriate for these data

$$y_{ij} = \mu + b_i + \epsilon_{ij}, i = 1, \ldots, 6, \quad j = 1, \ldots, 3, \tag{32}$$

where

- y_{ij} is the travel time of rail i at jth measurement.
- μ is the mean travel time across the population of rails being sampled.
- b_i is a random variable representing the deviation from the population mean travel time for the ith rail.
- ϵ_{ij} is a random variable representing the deviation in travel time for observation j on rail i from the mean travel time for rail i.

The b_i and ϵ_{ij} are modelled as independent, constant variance normally distributed random variables:

$$b_i \overset{\text{iid}}{\sim} N(0, \sigma_b^2), \qquad \epsilon_{ij} \overset{\text{iid}}{\sim} N(0, \sigma^2), \quad i = 1, \ldots, 6, \quad j = 1, \ldots, 3.$$

Here σ_b^2 represents the between rail variability and σ^2 represents the within rail variability. To reinforce: the b_i are called random effects because they are associated with the particular experimental units (rails in this case) that are selected at random from the population of interest. They are effects because they represent a deviation from an overall mean. That is, the 'effect' of choosing rail i is to shift the mean travel time from μ to $\mu + b_i$. Note that the parameters of model (32) are μ, σ_b^2, and σ^2 and that the number of parameters will always be three, irrespective of the number of rails in the experiment. Although the random effects b_i may behave like parameters, formally they are just another level of random variation in the model so we do not 'estimate' them as such. One does, however, form predictions $\hat{b}_i$ of the values of these random variables, given the observed data.

Let us now inspect some properties of model (32). The expected value of y_{ij} is

$$\mathbb{E}(y_{ij}) = \mathbb{E}(\mu + b_i + \epsilon_{ij}) = \mu.$$

For the variance of y_{ij} we have

$$\text{var}(y_{ij}) = \text{var}\mu + b_i + \epsilon_{ij} = \sigma_b^2 + \sigma^2.$$

For the correlation structure we get

$$\text{Cor}(y_{ij}, y_{kl}) = \begin{cases} 0, & i \neq k, \\ \frac{\sigma_b^2}{\sigma_b^2+\sigma^2}, & i = k, j \neq l, \\ 1, & i = k, j = l. \end{cases}$$

Observations from different rails ($i \neq k$) are uncorrelated while observations from the same rail ($i = k$) are correlated. You are possibly wondering why $\text{Cor}(y_{ij}, y_{il}) = \sigma_b^2/(\sigma_b^2+\sigma^2)$. Let's then work it out. First, remember that

$$\text{Cor}(y_{ij}, y_{il}) = \frac{\text{Cov}(y_{ij}, y_{il})}{\sqrt{\text{var}(y_{ij})\text{var}(y_{il})}}.$$

By definition,

$$\begin{aligned} \text{Cov}(y_{ij}, y_{il}) &= \mathbb{E}[(y_{ij} - \mathbb{E}[y_{ij}])(y_{il} - \mathbb{E}[y_{il}])] \\ &= \mathbb{E}[(\mu + b_i + \epsilon_{ij} - \mu)(\mu + b_i + \epsilon_{il} - \mu)] \\ &= \mathbb{E}[(b_i + \epsilon_{ij})(b_i + \epsilon_{il})] \\ &= \mathbb{E}[b_i^2 + b_i\epsilon_{il} + \epsilon_{ij}b_i + \epsilon_{ij}\epsilon_{il}] \\ &= \mathbb{E}[b_i^2] + \mathbb{E}[b_i\epsilon_{il}] + \mathbb{E}[\epsilon_{ij}b_i] + \mathbb{E}[\epsilon_{ij}\epsilon_{il}]. \end{aligned}$$

Because b_i and ϵ_{ij} are independent we have that $\mathbb{E}[\epsilon_{ij}b_i] = \mathbb{E}[\epsilon_{ij}]\mathbb{E}[b_i] = 0$. For the same reason, $\mathbb{E}[b_i\epsilon_{il}] = 0$. Further, $\text{var}(b_i) = \mathbb{E}[b_i^2] - (\mathbb{E}[b_i])^2 = \sigma_b^2$ and because $\mathbb{E}[b_i] = 0$ it trivially follows that $\mathbb{E}[b_i^2] = \sigma_b^2$. Finally, ϵ_{ij} and ϵ_{il} are independent and hence $\mathbb{E}[\epsilon_{ij}\epsilon_{il}] = \mathbb{E}[\epsilon_{ij}]\mathbb{E}[\epsilon_{il}] = 0$. Bringing this all together leads to $\text{Cov}(y_{ij}, y_{il}) = \sigma_b^2$. We have already concluded that $\text{var}(y_{ij}) = \text{var}(y_{il}) = \sigma_b^2 + \sigma^2$ and thus,

$$\text{Cor}(y_{ij}, y_{il}) = \frac{\text{Cov}(y_{ij}, y_{il})}{\sqrt{\text{var}(y_{ij})\text{var}(y_{il})}} = \frac{\sigma_b^2}{\sqrt{(\sigma_b^2+\sigma^2)(\sigma_b^2+\sigma^2)}} = \frac{\sigma_b^2}{\sigma_b^2+\sigma^2}.$$

The correlation within the same rail, $\sigma_b^2/(\sigma_b^2+\sigma^2)$, is also called the *intraclass correlation coefficient*. It is large if $\sigma_b^2 \gg \sigma^2$ meaning that observations from the same 'group' (here, rails) are much more similar than observations from different groups (rails). This nonzero correlation within the same rail is in contrast to the analogous fixed effects model, where all values are independent, because there, the b_is are parameters, that is, they are fixed, unknown quantities, and therefore also uncorrelated.

Now we fit the random effects model with the `lmer` function in package `lme4` (that needs to be installed!). We want to have a random effect per rail. This can be specified with the notation `(1 | Rail)` in the model formula. This means that the 'granularity' of the random effect is specified after the vertical bar 'l'. All observations sharing the same level of rail will get the same random effect b_i (in this case we know that there are three measurements per rail). The '1' means that we only want to have a so-called random intercept (b_i) per rail.

```
library(lme4)
rm <- lmer(travel ~ (1 | Rail), data = Rail)
```

As usual, the function summary gives an overview of the fitted model.

```
summary(rm)

## Linear mixed model fit by REML ['lmerMod']
## Formula: travel ~ (1 | Rail)
##    Data: Rail
##
## REML criterion at convergence: 122.2
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -1.61883 -0.28218  0.03569  0.21956  1.61438
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Rail     (Intercept) 615.31   24.805
##  Residual              16.17    4.021
## Number of obs: 18, groups:  Rail, 6
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)    66.50      10.17   6.538
```

From the summary we have that the estimates of the parameters are as follows

$$\hat{\mu} = 66.50, \quad \hat{\sigma}_b = 24.805, \quad \hat{\sigma} = 4.021.$$

Note that $\hat{\sigma}_b^2 \gg \hat{\sigma}^2$ meaning that travel times within rail are much more similar than travel times across rails (as we have already noticed from the plot of the data).

(Approximate) 95% confidence intervals for the parameters can be obtained with the function `confint` (the argument `oldNames` has to bet set to FALSE to get a nicely labelled output).

```
confint(rm, oldNames = FALSE)
```

```
##                           2.5 %    97.5 %
## sd_(Intercept)|Rail 13.929422 45.546746
## sigma                2.824557  6.377695
## (Intercept)         44.960664 88.039333
```

Hence, an approximate 95% confidence interval for the standard deviation σ_b is given by $(13.93, 45.55)$. We see that the estimate $\hat{\sigma}_b$ is therefore quite imprecise. The reason is that we only have six rails to estimate the standard deviation.

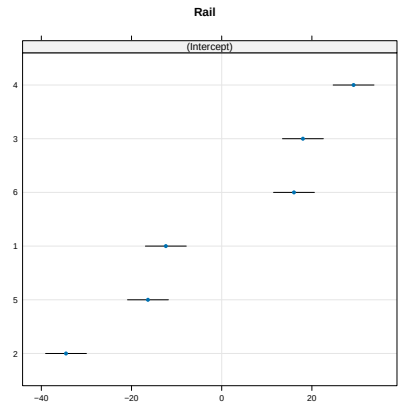
We can also get 'estimates' of the random effects b_i with the function `ranef`.

```
ranef(rm)
```

```
## $Rail
##   (Intercept)
## 2   -34.53091
## 5   -16.35675
## 1   -12.39148
## 6    16.02631
## 3    18.00894
## 4    29.24388
##
## with conditional variances for "Rail"
```

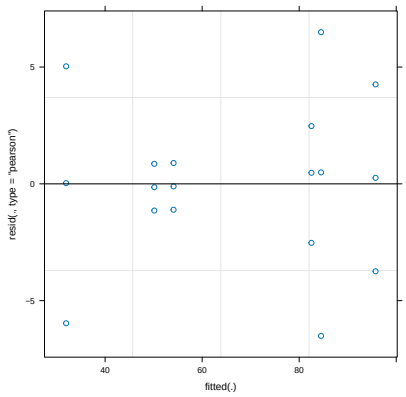
These can also be plotted along with confidence intervals.

```
library(lattice)
dotplot(ranef(rm, condVar = TRUE))

## $Rail
```

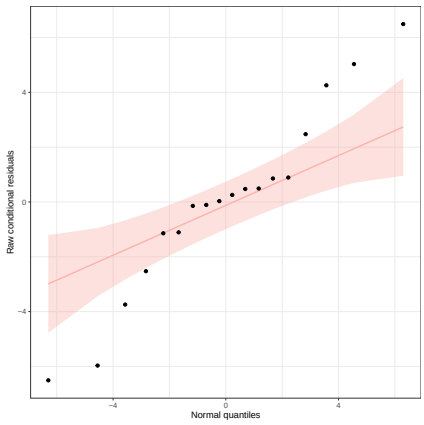


We should of course also check the model assumptions. Here, this includes in addition normality of the random effects, though this is hard to check with only six observations.

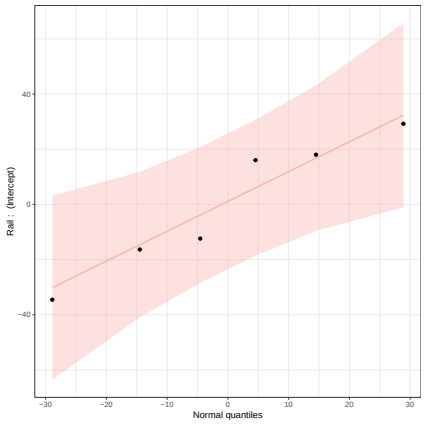
```
plot(rm)
```




```
library(redres)
plot_resqq(rm)
```




```
plot_ranef(rm)
```

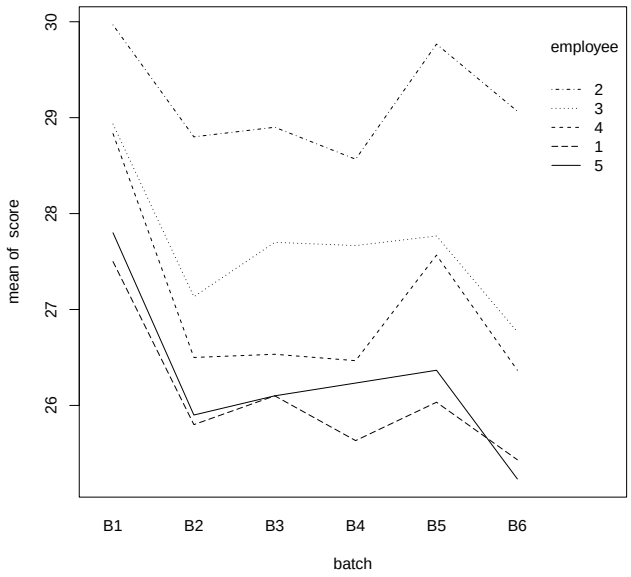


13.2 A random effects model involving a random interaction

Let's consider now an example involving two random factors and their (random) interaction. We will consider the following example: a large company randomly selected five employees and six batches of source material from the production process. The material from each batch was divided into 15 pieces which were randomized to the different employees such that each employee would build three test specimens from each batch. The response was the corresponding quality score. The goal was to quantify the different sources of variation.

As usual, we first load the data and get an overview.

```
book.url <- "https://stat.ethz.ch/~meier/teaching/book-anova"
quality <- readRDS(url(file.path(book.url, "data/quality.rds")))
str(quality)

## 'data.frame': 90 obs. of  3 variables:
##  $ employee: Factor w/ 5 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ batch   : Factor w/ 6 levels "B1","B2","B3",..: 1 1 1 2 2 2 3 3 3 4 ...
##  $ score   : num  27.4 27.8 27.3 25.5 25.5 26.4 26.9 26.3 25.1 25.6 ...

head(quality)

##   employee batch score
## 1        1    B1  27.4
## 2        1    B1  27.8
## 3        1    B1  27.3
## 4        1    B2  25.5
## 5        1    B2  25.5
## 6        1    B2  26.4

#interaction plot
with(quality, interaction.plot(x.factor = batch,
                               trace.factor = employee,
                               response = score))
```



There seems to be variation between different employees and between the different batches. The interaction does not seem to be very pronounced but for the sake of illustration we will include it in our model. Let y_{ijk} denote the observed quality score of the kth sample of employee i using material from batch j. The model we will be fitting is

$$y_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon_{ijk},$$

where α_i is the random (main) effect of employee i, β_j is the random (main) effect of batch, γ_{ij} is the random interaction term between employee i and batch j and $\epsilon_{ijk} \overset{\text{iid}}{\sim} N(0, \sigma^2)$ is the error term. For the random effects we have the usual assumptions

$$\alpha_i \overset{\text{iid}}{\sim} N(0, \sigma_\alpha^2), \quad \beta_j \overset{\text{iid}}{\sim} N(0, \sigma_\beta^2), \quad \gamma_{ij} \overset{\text{iid}}{\sim} N(0, \sigma_\gamma^2).$$

In addition we assume that all random effects and the error term are independent of each other.

What is the interpretation of the different terms? The random (main) effect of employee is the variability between different employees, and the random (main) effect of batch is the variability between different batches. The random interaction term can, for example, be interpreted as quality inconsistencies of employees across different batches.

Let us fit this model in R.

```
qm <- lmer(score ~ (1 | employee) + (1 | batch) + (1 | employee:batch),
           data = quality)
summary(qm)

## Linear mixed model fit by REML ['lmerMod']
## Formula: score ~ (1 | employee) + (1 | batch) + (1 | employee:batch)
##    Data: quality
##
## REML criterion at convergence: 167.5
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.8144 -0.5933 -0.1190  0.7073  2.8565
##
## Random effects:
```

```
##  Groups         Name        Variance Std.Dev.
##  employee:batch (Intercept) 0.02349  0.1533
##  batch          (Intercept) 0.51764  0.7195
##  employee       (Intercept) 1.54473  1.2429
##  Residual                   0.22655  0.4760
## Number of obs: 90, groups:  employee:batch, 30; batch, 6; employee, 5
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)  27.2478     0.6313   43.16
```

From the output we get $\hat{\sigma}^2_\alpha = 1.54$, $\hat{\sigma}^2_\beta = 0.52$, $\hat{\sigma}^2_\gamma = 0.02$ and $\hat{\sigma}^2 = 0.23$. Hence, total variance is $1.54 + 0.52 + 0.02 + 0.23 = 2.31$. We see that the largest contribution to the variance is variability between different employees which contributes to about $1.54/2.31 \approx 67\%$ of the total variance. These are all point estimates. Confidence intervals on the scale of the standard deviations can be obtained by calling `confint`.

```
confint(qm, oldNames = FALSE)
```

```
##                                    2.5 %     97.5 %
## sd_(Intercept)|employee:batch  0.0000000  0.3527999
## sd_(Intercept)|batch           0.4100059  1.4754427
## sd_(Intercept)|employee        0.6694340  2.4993883
## sigma                          0.4020266  0.5741431
## (Intercept)                   25.9189910 28.5765641
```

There is no statistical evidence that the random interaction term is really needed, as the corresponding confidence interval contains zero.

13.3 A mixed effects model involving both a fixed and a random factor effect

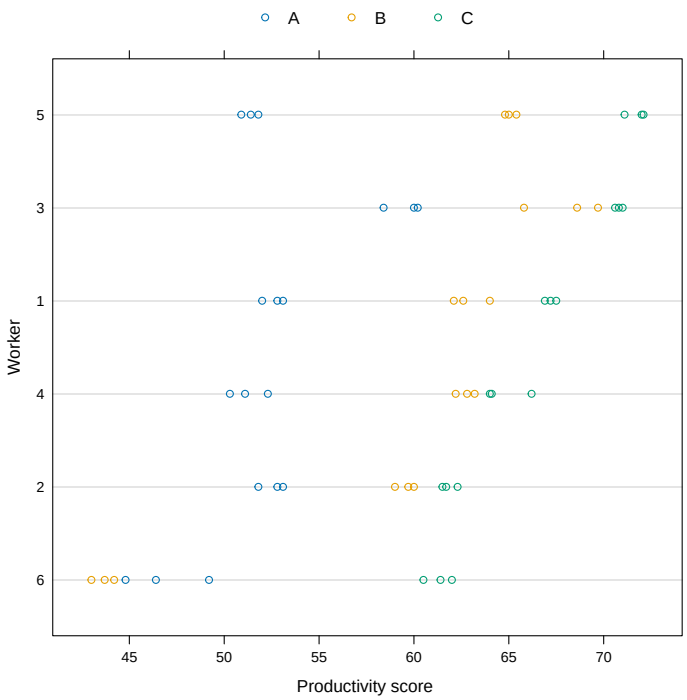
In practice, we often encounter models which involve both random and fixed effects. As already mentioned, these are designated by *mixed effects models*. To illustrate this class of models we will use the `Machines` dataset available in `nlme`, which contains data on an experiment to compare three brands of machines used in an industrial process. Six workers were chosen randomly among the employees of a factory to operate each machine three times. The response is an overall productivity score taking into account the number and quality of components produced. Let's look at the first rows of the dataset.

```
head(Machines)
```

```
## Grouped Data: score ~ Machine | Worker
##   Worker Machine score
## 1      1       A  52.0
## 2      1       A  52.8
## 3      1       A  53.1
## 4      2       A  51.8
## 5      2       A  52.8
## 6      2       A  53.1
```

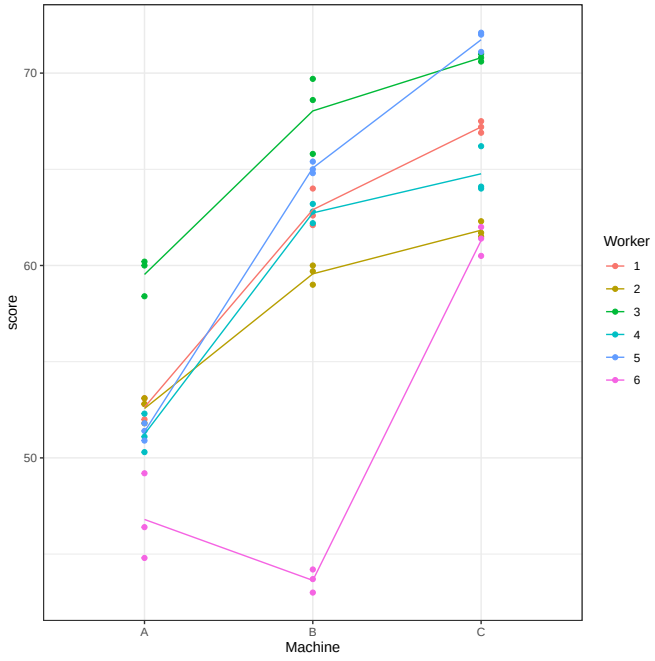
In the figure below we can see that there are strong indications of differences between machines and also some indications of differences between workers. We further note that there is very little variability in the productivity score for the same worker using the same machine. In addition, on average, productivity is largest on machine C, followed by B, and A.

```
plot(Machines)
```



The below plot shows that there may be an interaction between worker and machine.

```
## technical details for shorter output:
class(Machines) <- "data.frame"
Machines[, "Worker"] <- factor(Machines[, "Worker"], levels = 1:6,
                               ordered = FALSE)
require(ggplot2)
ggplot(Machines, aes(x = Machine, y = score, group = Worker, col = Worker)) +
  geom_point() + stat_summary(fun = mean, geom = "line") + theme_bw()
```



Let us now posit a model on this data. The goal is to make inference about the specific machines at hand, this is why we treat machine as a fixed effect. Because the workers represent a random sample from the population of interest, their effect will be considered a random effect. We will include an interaction between worker and machine, and an interaction effect between a random (here, worker) and a fixed effect (here, machine) is treated as a random effect. Let y_{ijk} denote the observed kth productivity score of worker j on machine i. We use the model

$$y_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon_{ijk},$$

where α_i is the fixed effect of machine, β_j is the random effect of worker j, and γ_{ij} is the corresponding random interaction effect. We assume that all random effects are normally distributed, that is,

$$\beta_j \overset{\text{iid}}{\sim} N(0, \sigma_\beta^2), \quad \gamma_{ij} \overset{\text{iid}}{\sim} N(0, \sigma_\gamma^2).$$

For the error term we assume as always $\epsilon_{ijk} \overset{\text{iid}}{\sim} N(0, \sigma^2)$. In addition, all random terms are assumed to be independent.

We now fit the model through the `lmer` function. As `lme4` does not calculate p-values for the fixed effects, we use the package `lmerTest` instead. Technically speaking, `lmerTest` uses `lme4` to fit the model and then adds some statistical tests, i.e., p-values for the fixed effects, to the output. We can also do an F test through the `anova` (but from the `lmerTest` package) function.

```
library(lmerTest)
fit.machines <- lmer(score ~ Machine + (1 | Worker) + (1 | Worker:Machine),
                     data = Machines)
summary(fit.machines)

## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: score ~ Machine + (1 | Worker) + (1 | Worker:Machine)
##    Data: Machines
##
## REML criterion at convergence: 215.7
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.26959 -0.54847 -0.01071  0.43937  2.54006
##
## Random effects:
##  Groups         Name        Variance Std.Dev.
##  Worker:Machine (Intercept) 13.9095  3.7295
##  Worker         (Intercept) 22.8584  4.7811
##  Residual                    0.9246  0.9616
## Number of obs: 54, groups:  Worker:Machine, 18; Worker, 6
##
## Fixed effects:
##             Estimate Std. Error     df t value Pr(>|t|)
## (Intercept)   52.356      2.486  8.522  21.062 1.20e-08 ***
## MachineB       7.967      2.177 10.000   3.660  0.00439 **
## MachineC      13.917      2.177 10.000   6.393 7.91e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##          (Intr) MachnB
## MachineB -0.438
## MachineC -0.438  0.500

anova(fit.machines)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##         Sum Sq Mean Sq NumDF DenDF F value    Pr(>F)
## Machine 38.051  19.025     2    10  20.576 0.0002855 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

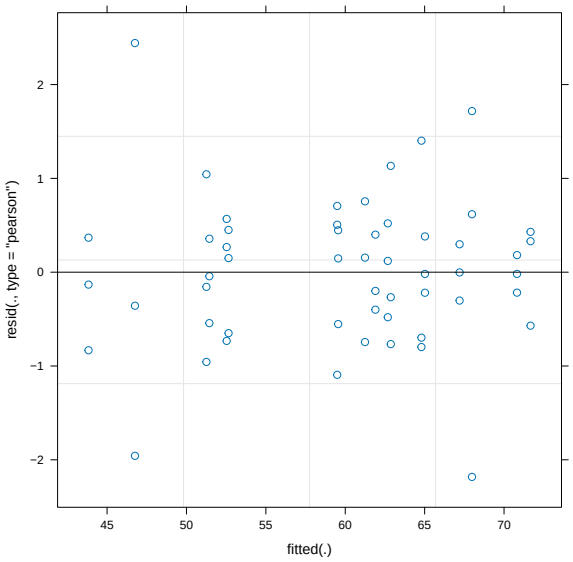
The fixed effect of machine is significant. We further have that $\hat{\sigma}_\beta = 4.78$ (between worker variability), $\hat{\sigma}_\gamma = 3.73$ and $\hat{\sigma} = 0.96$ (within-worker variability). As before, confidence intervals can also be easily calculated.

```
confint(fit.machines, oldNames = FALSE)
```

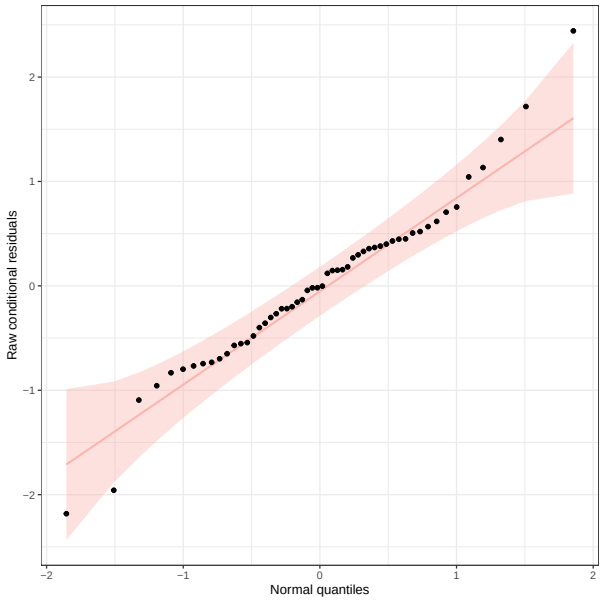
```
##                                    2.5 %    97.5 %
## sd_(Intercept)|Worker:Machine  2.3528037  5.431503
## sd_(Intercept)|Worker          1.9514581  9.410584
## sigma                          0.7759507  1.234966
## (Intercept)                   47.3951611 57.315949
## MachineB                       3.7380904 12.195243
## MachineC                       9.6880904 18.145243
```

We can do the residual analysis as outlined before.

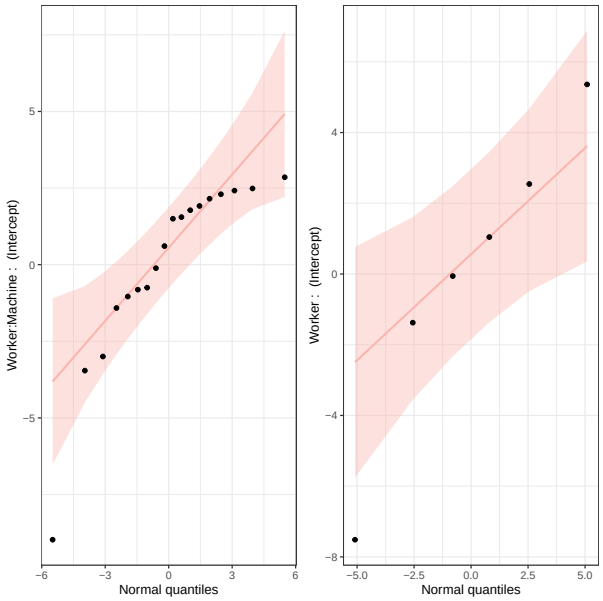
```
plot(fit.machines)
```




```
plot_resqq(fit.machines)
```




```
plot_ranef(fit.machines)
```



All seems ok! In order to better understand the mixed model, we check what happens if we would fit a purely fixed effects model here.

```
fit_machines_fe <- lm(score ~ Machine*Worker, data = Machines)
anova(fit_machines_fe)

## Analysis of Variance Table
##
## Response: score
##                Df  Sum Sq Mean Sq F value    Pr(>F)
## Machine         2 1755.26  877.63  949.17 < 2.2e-16 ***
## Worker          5 1241.89  248.38  268.63 < 2.2e-16 ***
## Machine:Worker 10  426.53   42.65   46.13 < 2.2e-16 ***
## Residuals      36   33.29    0.92
```

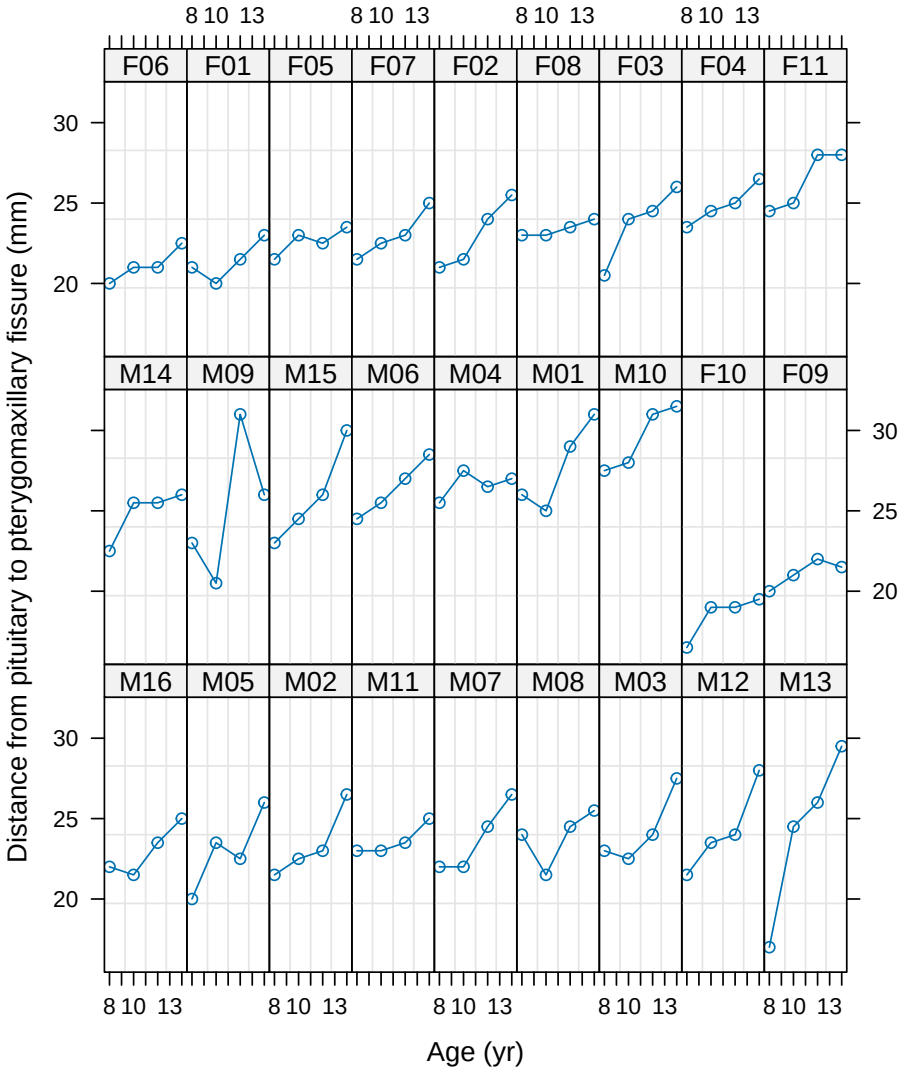
```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The machine effect is much more significant. This is because in the fixed effects model, the main effect of machine makes a statement about the average machine effect of these 6 specific workers and not about the population average.

13.4 A mixed effects model involving both a (continuous) fixed and a random effect

A common application of random-effects analysis with continuous predictors is in modeling *growth curve* data: the results on different subjects of repeated measurements of some characteristic over time. A classic example of such data is a set of measurements of the distance from the pituitary gland to the pterygo-maxillary fissure taken every two years from 8 years of age until 14 years of age on a sample of 27 children: 16 males and 11 females. The data are available as `Orthodont` in the `nlme` package were collected by orthodontists from x-rays of the children's skulls.

```
plot(Orthodont)
```



It seems that there are qualitative differences between boys and girls in their growth patterns for this measurement. For the sake of simplicity, let us restrict our modeling to the data from the female subjects only.

```
OrthoFem <- Orthodont[ Orthodont$Sex == "Female", ]
head(OrthoFem)

## Grouped Data: distance ~ age | Subject
##    distance age Subject    Sex
## 65     21.0   8     F01 Female
## 66     20.0  10     F01 Female
## 67     21.5  12     F01 Female
## 68     23.0  14     F01 Female
## 69     21.0   8     F02 Female
## 70     21.5  10     F02 Female
```

As a matter of illustration, let us use a model the following model

$$y_{ij} = \mu + \beta \text{age}_{ij} + b_i + \epsilon_{ij},$$

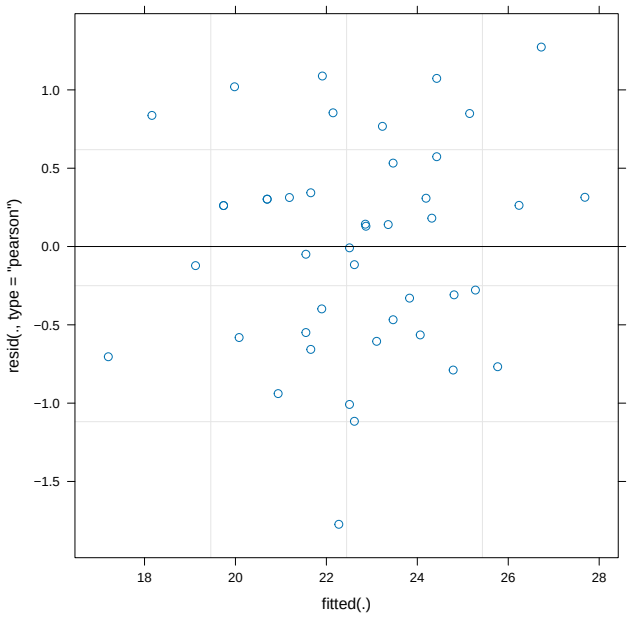
where y_{ij} is the distance for girl i measured at age j, βage_{ij} is the fixed effect of age and b_i is the girl/subject random effect. Fitting a mixed-effects model to these data allows to make inferences about the fixed effects, which represent average characteristics of the population represented by these subjects, and the variability amongst girls.

```
fit_girls <- lmer(distance ~ age + (1|Subject), data = OrthoFem)
summary(fit_girls)

## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: distance ~ age + (1 | Subject)
##    Data: OrthoFem
##
## REML criterion at convergence: 141.2
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.2736 -0.7090  0.1728  0.4122  1.6325
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Subject  (Intercept) 4.2786   2.068
##  Residual             0.6085   0.780
## Number of obs: 44, groups:  Subject, 11
##
## Fixed effects:
##             Estimate Std. Error       df t value Pr(>|t|)
## (Intercept) 17.37273    0.85874 27.57177  20.230  < 2e-16 ***
## age          0.47955    0.05259 32.00000   9.119 2.06e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##     (Intr)
## age -0.674

confint(fit_girls, oldNames = FALSE)

##                              2.5 %     97.5 %
```

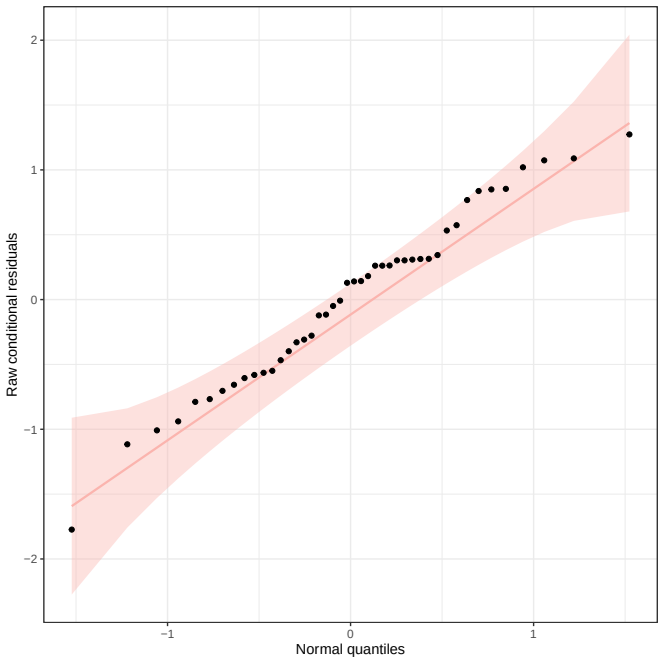
```
## sd_(Intercept)|Subject  1.3356536  3.2373519
## sigma                   0.6143927  0.9985162
## (Intercept)            15.6891967 19.0562579
## age                     0.3750181  0.5840728
```

We see that the fixed effect of age is significant and that the variability between girls is higher than within girls. To conclude, let's look at residual plots.

```
plot(fit_girls)
```




```
plot_resqq(fit_girls)
```



```
plot_ranef(fit_girls)
```