# Some basic statistics with R

Ramon Diaz-Uriarte
Dept. Biochemistry, Universidad Autónoma de Madrid
Instituto de Investigaciones Biomédicas "Alberto Sols" (UAM-CSIC)
Madrid, Spain*
http://ligarto.org/rdiaz

2017-01-20 (Release 1.2: Rev: f4869d8)

## Contents



*ramon.diaz@iib.uam.es, rdiaz02@gmail.com

# 1 License and copyright



# 2 Introduction

This document is a brief reviews of statistical approaches for comparing two groups (t-tests et al.) and linear models (anovas, regressions, etc). It assumes you know a little bit of R. You will probably need to install several additional packages as you go along; you will see a something like `library(something)` or `require(something)`.

## 2.1 The PDF and the code

The primary output of this document is a PDF. All the original files for the document are available (again, under a Creative Commons license —see section 1) from https://github.com/rdiaz02/R-basic-stats. (Note that in the github repo you will not see the PDF, HTML, or R files, since those are derived from the Rnw file).

For many commands I do not show the output (e.g., because it would just provide boring and space-filling output). However, make sure you type and understand it. You can copy and paste, of course, but I strongly suggest you type the code and change it, etc.

## 2.2 Warning: eternally provisional

This file can get changed often. When asking questions in class or in the forum, refer to the section numbers AND section names (these change less than the page numbers).

*Part I*

# *Comparing two groups*

## 3 Introduction to comparing two groups

Someone in your lab has measured the expression of several genes from a set of patients with and without cancer. You are in charge of looking at the data and answering the question "Does the expression of the genes differ between patients with and without cancer?".

### 3.1 Files we will use

- This one
- `P53.txt`
- `MYC.txt`
- `BRCA2.txt`

## 4 Types of data

We need to get this out of the way, as we will refer to it frequently. Data can be measured in different scales. From "less information to more information" we can organize scales this way:

**Nominal or categorical scale** We use a scale that simply differentiates different classes. For instance we can classify some objects around here, "computer", "blackboard", "pencil", and we can give numbers to them (1 to computer, 2 to blackboard, etc) but the numbers have no meaning per se.

**Binary** data are in a nominal scale with only two classes: dead or alive (and we can give a 0 or a 1 to either), male or female, etc.

Lots of biological data are in a nominal scale. For instance, suppose you look at the types of repetitive elements in the genome, and give a 1 to SINEs, a 2 to LINEs, etc. Or you number the aminoacids from 1 (alanine) to 20 (valine). You can of course count how many are of type 1 (how many are alanines), etc, but it would make no sense to do averages and say "your average AA composition is 13.5".

**Ordinal scale** The data can be ordered in the sense that you can say that something is larger or smaller than something else. For instance, you can rank your preference for food as:

"chocolate > jamon serrano > toasted crickets > liver". You might assign the value 1 to chocolate (your most preferred food) and a 4 to liver (the least preferred) but differences or ratios between those numbers have no meaning.
Some measurements in biology are of these kind. Name a few?

**Interval or ratio scale** You can take differences and ratios, and they do have meaning[1]. If a subject has a value of 6 for the expression of gene PTEN, another a value of 3, and another a value of 1, then the first has six times more RNA of PTEN than the last, and two times more than the second.

We will try to be careful. With nominal data we will always try to keep them as "things without numbers", so that we make no mistakes (i.e., keep the aminoacids names, not just a bunch of 1 to 20). Ordinal scale are trickier, and we will need to think about the type of data and the analyses.

# 5 Looking at the data: plots

We first need to import the data. Make sure you name it sensibly; for instance, dp53:

```
dp53 <- read.table("P53.txt", header = TRUE)
```

The first step ever is to look at the data. In fact, here we can look at all the original data. So go take a look at the data.

## 5.1 Plots to do

For all except the trivially tiniest datasets we want to use graphics. Make sure you do the following plots:

- Histogram for each gene, using condition ("cond") as the conditioning or grouping variable ("Plot by:").
- Boxplot, using condition ("cond") as the conditioning or grouping variable ("Plot by:").
- Plot of means (and make sure you get nicer axes labels).
- Stripchart, and make sure you use "jitter", not "stack": can you tell for which one of the variables this matters a lot?
- Density plots ("Density estimates")

We will load a few packages we need:

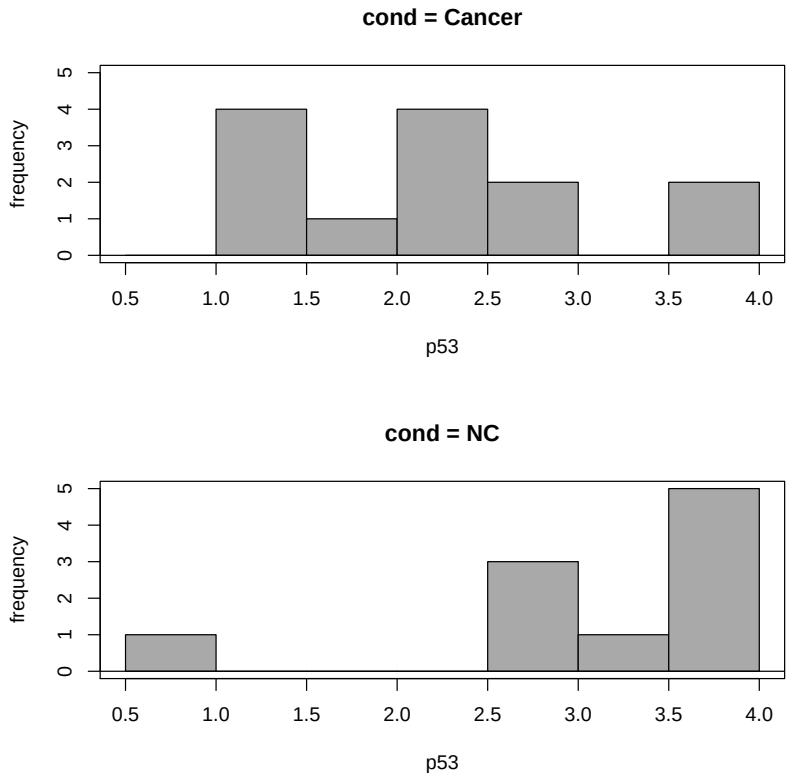
```
library(car)
library(RcmdrMisc)
```

```
## Loading required package: sandwich
```

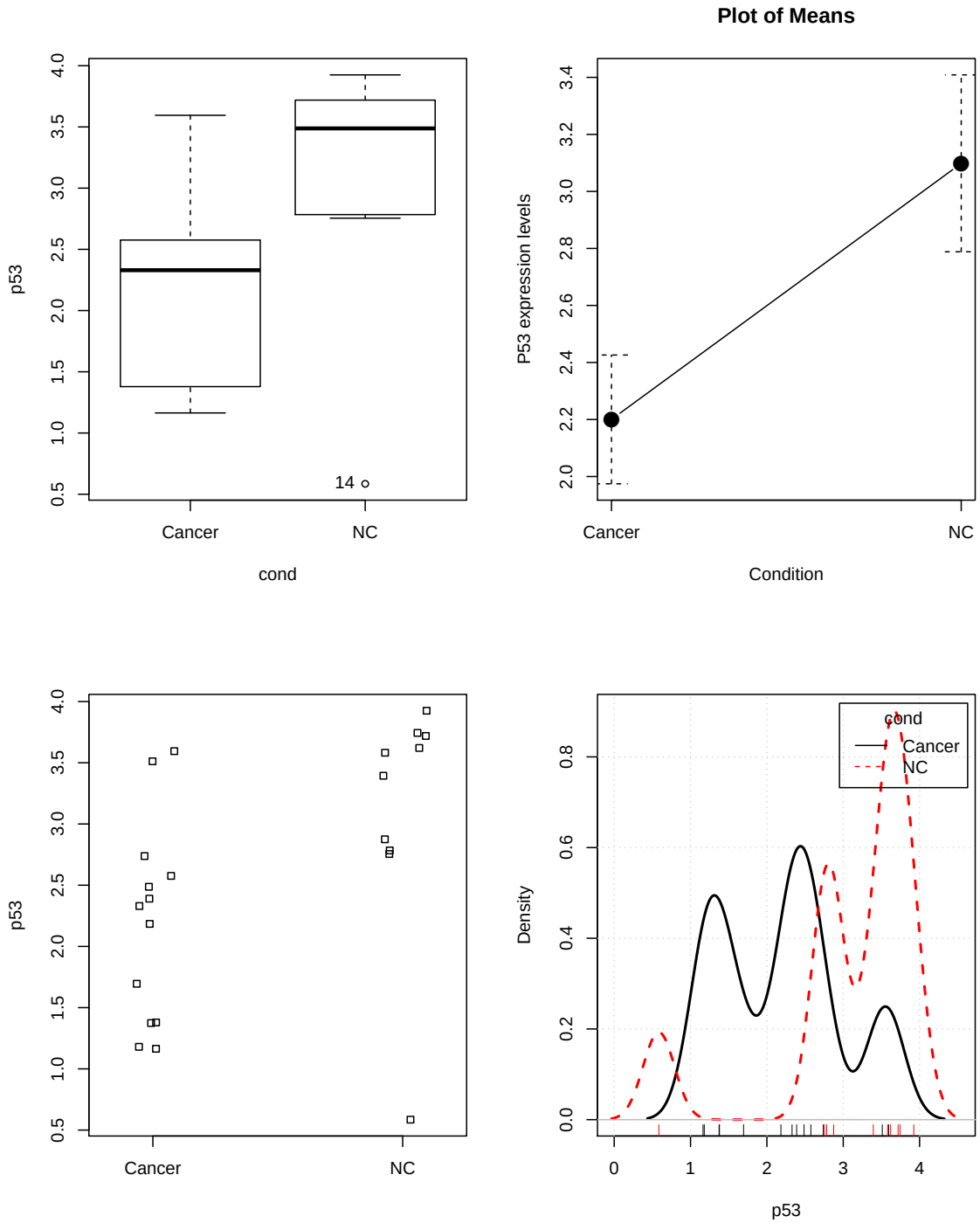
To get you going, I show all those for p53.

```
with(dp53, Hist(p53, groups = cond, col = "darkgray"))
```

[1] Some authors make a distinction between ratio and interval scales; I won't. The only difference is that ratio scales have a natural zero.




```
op <- par(mfrow = c(2, 2)) ## to show 2 by 2 on the same figure
Boxplot(p53 ~ cond, data = dp53, id.method = "y")
plotMeans(dp53$p53, dp53$cond, error.bars = "se",
  xlab = "Condition", ylab = "P53 expression levels")
stripchart(p53 ~ cond, vertical = TRUE, method = "jitter",
  ylab = "p53", data = dp53)
densityPlot(p53 ~ cond, data = dp53, bw = "SJ", adjust = 1,
  kernel = "gaussian")
```




```
par(op)
```

Note that I used the `Boxplot`, but we could have used `boxplot`. The first is from *car* and provides added functionality. The `op <- par(mfrow = c(1,2))` and `par(op)` are a minor detail to restore options as they were.

Likewise, I have used `Hist` and `plotMisc` from *RcmdrMisc*. They are not strictly needed (you can get these plots by other means), but they are extremely convenient.

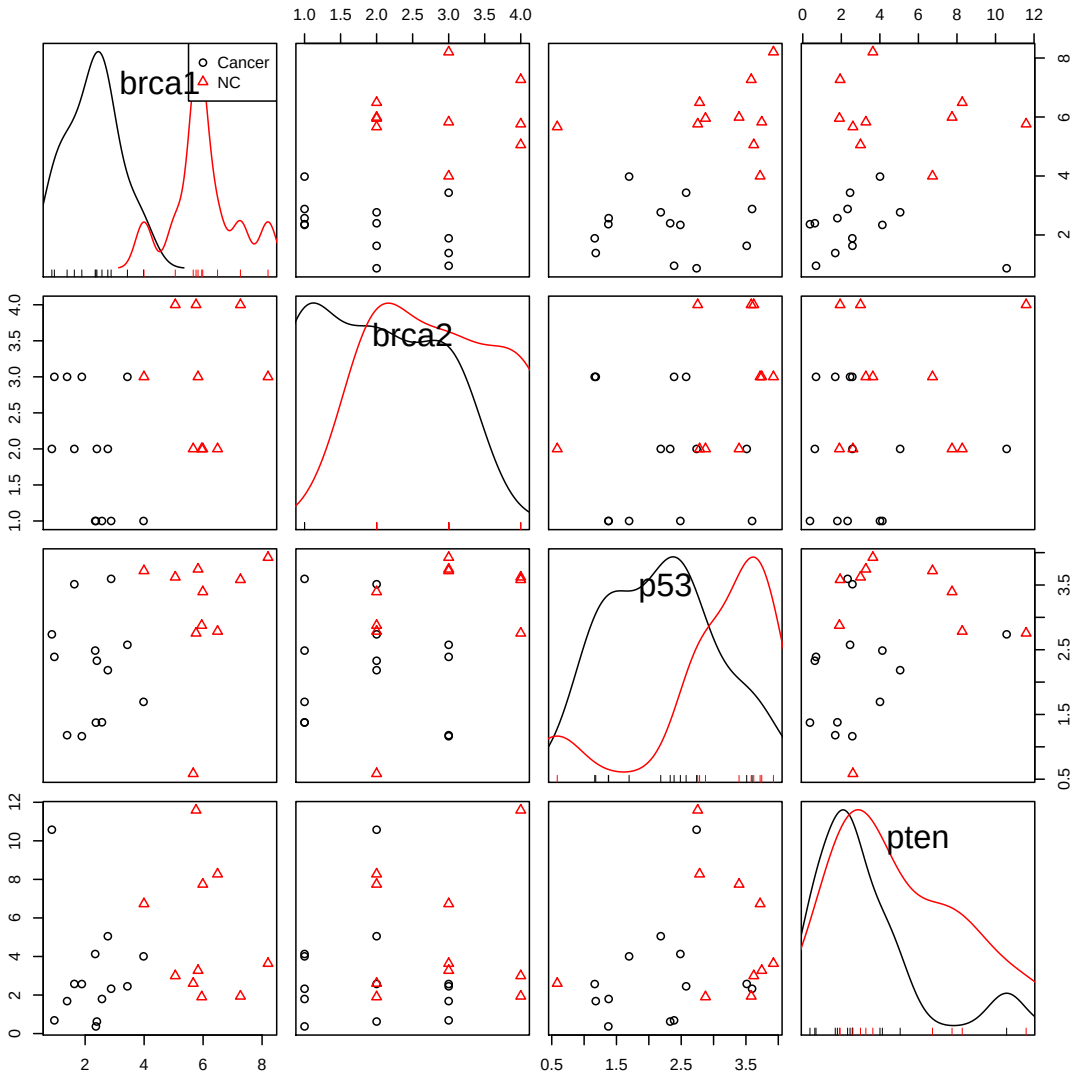
## 5.2 What are the plots telling you?

Now, think about what these plots tell you:

1. The boxplots, is the "boxplot by group" more or less useful than the built-in one?
2. When was the stripchart specially useful? Could you see anything strange for brca2 without the stripchart?
3. Eye balling the plots, what variables do you think show differences between the two conditions? Wait! Think about at least:
   (a) Differences in mean/median
   (b) Differences in dispersion (variance, IQR, etc)
4. Similar question again: what genes look like they have differential expression between the cancer and non-cancer patients?
5. Are density plots reasonable in these cases?

### 5.3 Relations between variables

We will focus on comparing two groups. But we have several variables (genes). An obvious thing to do is to look at how they are related AND display the different (two, in this case) groups.




We will not pursue this any further. But you know you can and probably want to look at these kinds of plots routinely.

## 6 Comparing two groups with a t-test

Let's start with p53.

```
t.test(p53 ~ cond, data = dp53)

##
##  Welch Two Sample t-test
##
## data:  p53 by cond
## t = -2.3402, df = 17.402, p-value = 0.03142
```

```
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.70635155 -0.08983306
## sample estimates:
## mean in group Cancer     mean in group NC
##             2.200308             3.098400
```

- What is this test for?
- Do you remember what the formula for the t-statistic look like? And why should this matter?
- What are the options given?
  - Equal variances? Does it make a difference? Any simple hint of whether you are using Welch's test or the equal-variances one?
  - Alternative hypothesis? One-tailed vs. two-tailed.
- Do results agree with the figures? What figures?
- Oh, in fact: can we interpret the results?

(We will spend time in class making sure all this is understood if you have forgotten your stats classes).

Repeat the above with brca1.

## 6.1 Assumptions of the t-test

One key assumption is **independence** of the data. This is the case for the t-test but is also the case for many statistical tests. We return to this below several times because it is a **crucial** assumption (e.g., section 12 and 23). Lack of independence is a serious and pervasive problem (and one form of non-independence is also referred to as "pseudoreplication" [2]).

When comparing two means, **equality of variances** is also important. But detecting differences in variances is complicated. Two practical solutions are: Welch's test (the default in R) and using transformations (see also section 7). However, think about the meaning of a comparison of means when variances are hugely different: do you really want to do that?

What about normality? It matters, but not as much, especially as sample size gets larger. Deviations from normality because of skewness (asymmetry) can have a large effect. Deviations because of kurtosis (tails heavier than the normal) have very minor effects. That is why we often say "data are sufficiently close to normality". And in the "sufficiently close" we worry mainly about asymmetries. And things tend to get "sufficiently close" as sample size grows larger (a consequence of the central limit theorem); how large is large enough? Oh, it depends on how far the distribution of the data is from the normal distribution, but often 10 is large enough, and 50 is generally well large enough (but in certain cases 100 might not be even close to large enough).

Of course, although this should be obvious: when we talk about symmetry, normality, etc, we are talking about the data distribution of **each group separately**.

Finally, **outliers** can be a serious concern (by the way, outliers, or potential outliers —under some definition of outlier— get flagged by function `Boxplots` in R). In general, points very far from the rest of the points will have severe effects on the value computed for the mean (not the median —this is also related to why nonparametric procedures can be more robust). What to do, however, is not obvious. An outlier might be the consequence of an error in data recording. But it might also be a perfectly valid point and it might actually be the "interesting stuff". Sometimes

[2] A major paper, a long while ago, in the journal *Ecological Monographs* by Hurlbert dealt with this and made "pseudoreplication" a well known term.

people carry out (and, of course, **should explicitly report**) analysis with and without the outlier; sometimes the same qualitative conclusions are reached, sometimes not. Please, think carefully about what an outlier is before proceeding with your analysis but do not get into the habit of automatically getting rid of potential outliers. And whatever you do, you should report it.

The book by Rupert Miller "Beyond Anova" (Chapman and Hall, 1997) contains a great discussion of assumptions, consequences of deviations from them, what to do, etc.

### 6.2 Ideas that should be clear from the t-test part

A few ideas that should be clear after this section:

1. The difference between a sample and a population
2. What a statistic is
3. What a t-statistic is
4. That statistics have distributions
5. The difference between standard deviation and standard error
6. That sampling introduces variability
7. That some procedures "ask more from the data" (e.g., interval data)
8. p-values
9. Null hypothesis
10. Distribution of the statistic under the null hypothesis
11. The logic behind a statistical test
12. The difference between estimation and hypothesis testing

If any one of the above is not clear, please ask it in class.

## 7 Data transformations

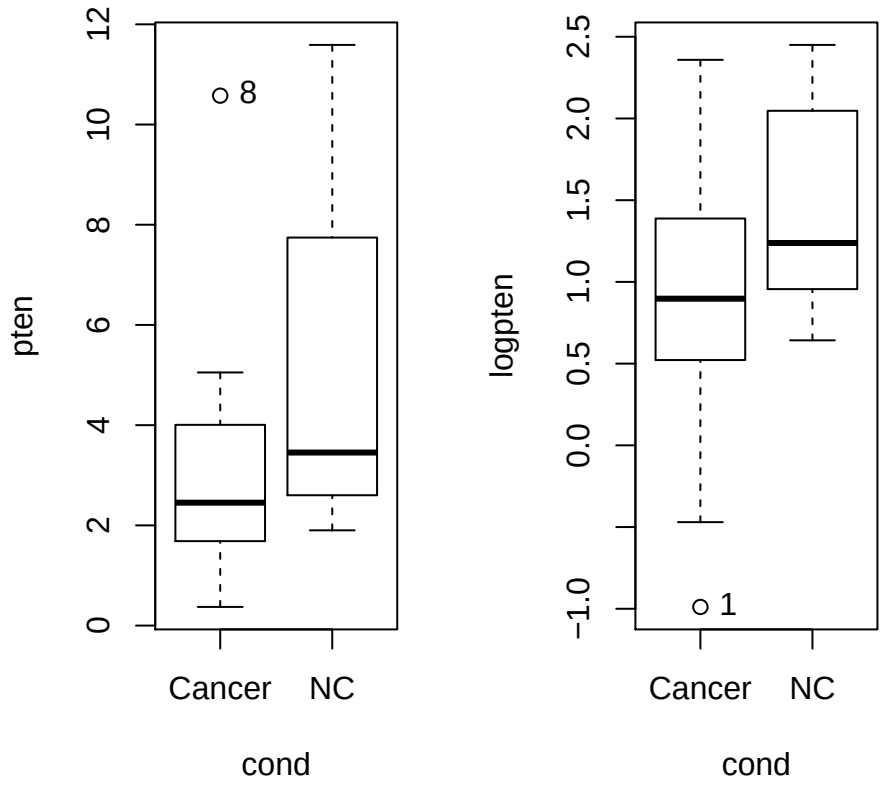
(Look at this section briefly if we do not have a lot of time. Many things will be clearer after we play with transformations in linear models.)

Let us look at the boxplot for "pten". What does it look like? Take the log transformation: create a new variable.

```
dp53$logpten <- log(dp53$pten)
```

Now, plot the log of pten. And do t-tests of the original and log-transformed variable.

```
op <- par(mfrow = c(1, 2))
Boxplot(pten ~ cond, data = dp53, id.method = "y")
Boxplot(logpten ~ cond, data = dp53, id.method = "y")
par(op)
```




Do we expect multiplicative or additive effects? (With a two-group comparisons it isn't always easy to think this, but can be crucial in regression; we will see it later).

```
t.test(pten ~ cond, data = dp53)
```

```
##
##  Welch Two Sample t-test
##
## data:  pten by cond
## t = -1.6219, df = 17.121, p-value = 0.1231
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -4.7824084  0.6240391
## sample estimates:
## mean in group Cancer     mean in group NC
##             2.988615             5.067800
```

```
t.test(logpten ~ cond, data = dp53)
```

```
##
##  Welch Two Sample t-test
##
## data:  logpten by cond
## t = -2.1112, df = 20.87, p-value = 0.04699
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.36940227 -0.01007187
```

```
## sample estimates:
## mean in group Cancer     mean in group NC
##            0.7451483            1.4348854
```

# 8 Paired tests

Load now the data set called `MYC.txt`. As before, give it a reasonable name.

```
dmyc <- read.table("MYC.txt", header = TRUE)
```

Look at the data. Anything interesting?

There are actually two observations per subject, one from tumor tissue and one from non-tumor tissue. This is a classical setting that demands a paired t-test. Why? When answering this question think about the idea of using each subject as its own control.

How can we check the above?

```
all(with(dmyc, table(id, cond)) == 1)
```

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

## 8.1 Paired t-test

Let's do the paired t-test.

```
myc.cancer <- dmyc$myc[dmyc$cond == "Cancer"]
myc.nc <- dmyc$myc[dmyc$cond == "NC"]
t.test(myc.nc, myc.cancer, paired = TRUE)
```

```
##
##  Paired t-test
##
## data:  myc.nc and myc.cancer
## t = 4.079, df = 11, p-value = 0.001823
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.432056 1.444777
## sample estimates:
## mean of the differences
##               0.9384167
```

Of course, this **crucially assumes** that the data are ordered the same way by patient: the first myc.cancer is the same patient as the same myc.nc, etc. This is the case in our data, but need not be. We can check it:

```
dmyc
```

```
##       myc   cond         id
## 1  38.289 Cancer krupfebolw
## 2  76.188 Cancer tuxydrhpqa
## 3  24.621 Cancer dcwzqfvxno
## 4  54.079 Cancer vcjmutngdo
## 5  43.832 Cancer gdqxtljvuf
## 6   0.300 Cancer idgcwfblvz
## 7  31.055 Cancer jtqbflruwv
## 8  58.402 Cancer amcjgskyfu
## 9  29.723 Cancer qwurdtxljs
## 10  6.190 Cancer ztnqhxgwfs
## 11 52.626 Cancer jtrymxigds
```

```
## 12 22.089 Cancer cfxzokirqa
## 13 39.634     NC krupfebolw
## 14 78.361     NC tuxydrhpqa
## 15 24.396     NC dcwzqfvxno
## 16 53.902     NC vcjmutngdo
## 17 44.679     NC gdqxtljvuf
## 18  1.675     NC idgcwfblvz
## 19 32.260     NC jtqbflruwv
## 20 59.427     NC amcjgskyfu
## 21 30.300     NC qwurdtxljs
## 22  6.030     NC ztnqhxgwfs
## 23 54.494     NC jtrymxigds
## 24 23.497     NC cfxzokirqa
```

However, to ensure that order is OK we could have pre-ordered the data by patient ID and by condition within patient (this second step isn't really needed in this case, but is in general):

```
dmycO <- dmyc[order(dmyc$id, dmyc$cond), ]
dmycO
```

```
##       myc   cond         id
## 8  58.402 Cancer amcjgskyfu
## 20 59.427     NC amcjgskyfu
## 12 22.089 Cancer cfxzokirqa
## 24 23.497     NC cfxzokirqa
## 3  24.621 Cancer dcwzqfvxno
## 15 24.396     NC dcwzqfvxno
## 5  43.832 Cancer gdqxtljvuf
## 17 44.679     NC gdqxtljvuf
## 6   0.300 Cancer idgcwfblvz
## 18  1.675     NC idgcwfblvz
## 7  31.055 Cancer jtqbflruwv
## 19 32.260     NC jtqbflruwv
## 11 52.626 Cancer jtrymxigds
## 23 54.494     NC jtrymxigds
## 1  38.289 Cancer krupfebolw
## 13 39.634     NC krupfebolw
## 9  29.723 Cancer qwurdtxljs
## 21 30.300     NC qwurdtxljs
## 2  76.188 Cancer tuxydrhpqa
## 14 78.361     NC tuxydrhpqa
## 4  54.079 Cancer vcjmutngdo
## 16 53.902     NC vcjmutngdo
## 10  6.190 Cancer ztnqhxgwfs
## 22  6.030     NC ztnqhxgwfs
```

```
myc.cancer <- dmycO$myc[dmycO$cond == "Cancer"]
myc.nc <- dmycO$myc[dmycO$cond == "NC"]
t.test(myc.nc, myc.cancer, paired = TRUE)
```

```
##
##  Paired t-test
##
## data:  myc.nc and myc.cancer
```

```
## t = 4.079, df = 11, p-value = 0.001823
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.432056 1.444777
## sample estimates:
## mean of the differences
##               0.9384167
```

## 8.2 Reshaping the data for a paired t-test

### 8.2.1 The shape of the data

Often, when you know you are only going to do a paired test you can organize your data in a table structure like the one in Table 1:

| SubjectID | Tumor | Non-Tumor |
|---|---|---|
| pepe | 23 | 45 |
| maria | 29 | 56 |
| ... | ... | ... |

**Table 1** – Paired data in a "unstacked" shape/format.

That looks ok, but is really a very limiting format for the data. Think about what you would do if you had additional "tumor" and "non-tumor" data per subject, or you had additional covariates for each measure, etc. That is why we often want to have the data in a "stacked" format as in Table 2:

| SubjectID | Myc | Condition |
|---|---|---|
| pepe | 23 | tumor |
| pepe | 45 | nontumor |
| maria | 29 | tumor |
| maria | 56 | nontumor |
| ... | ... | ... |

**Table 2** – Paired data in a "stacked" shape/format.

This is arguably a much more useful way of storing the data for most general analyses and we can keep adding additional information if we need to. Whole papers have been written about these issues[3], but we will stop here.

### 8.2.2 Reshaping our data

So that is what we want: the values of myc of the same subject to be on the same row. We will do this reshaping in several different ways.

There are many alternative approaches for reshaping the data directly from the command line. Many, many, many[4]. And yes, it is unavoidable that this will get a little bit complicated. But manipulating and reshaping data must be done with a lot of care. You do not need to remember the details here, you just need to be aware of the issues.

[3] For example, Wickham's "Tidy data", in the *Journal of Statistical Software*: http://www.jstatsoft.org/v59/i10

[4] For instance: http://www.cookbook-r.com/Manipulating_data/Converting_data_between_wide_and_long_format/ or https://rpubs.com/bradleyboehmke/data_wrangling; these are heavily biased towards the "Hadley's way".

### 8.2.3 Reshaping with `unstack`

We will first use the R function `unstack`. This is the natural reverse operation of "stacking".

```
unstack(x = dmyc, form = myc ~  cond)
```

```
##    Cancer     NC
## 1  38.289 39.634
## 2  76.188 78.361
## 3  24.621 24.396
## 4  54.079 53.902
## 5  43.832 44.679
## 6   0.300  1.675
## 7  31.055 32.260
## 8  58.402 59.427
## 9  29.723 30.300
## 10  6.190  6.030
## 11 52.626 54.494
## 12 22.089 23.497
```

That seems to do it but . . . how can you be sure each row corresponds to the same subject? We are counting on it to work, but it might not if the data had not been ordered by subject id. (See also section 8.1). So we will be explicit here, ordering by condition and then id:

```
dmyc0 <- dmyc[order(dmyc$cond, dmyc$id), ]
dmyc0
```

```
##       myc   cond         id
## 8  58.402 Cancer amcjgskyfu
## 12 22.089 Cancer cfxzokirqa
## 3  24.621 Cancer dcwzqfvxno
## 5  43.832 Cancer gdqxtljvuf
## 6   0.300 Cancer idgcwfblvz
## 7  31.055 Cancer jtqbflruwv
## 11 52.626 Cancer jtrymxigds
## 1  38.289 Cancer krupfebolw
## 9  29.723 Cancer qwurdtxljs
## 2  76.188 Cancer tuxydrhpqa
## 4  54.079 Cancer vcjmutngdo
## 10  6.190 Cancer ztnqhxgwfs
## 20 59.427     NC amcjgskyfu
## 24 23.497     NC cfxzokirqa
## 15 24.396     NC dcwzqfvxno
## 17 44.679     NC gdqxtljvuf
## 18  1.675     NC idgcwfblvz
## 19 32.260     NC jtqbflruwv
## 23 54.494     NC jtrymxigds
## 13 39.634     NC krupfebolw
## 21 30.300     NC qwurdtxljs
## 14 78.361     NC tuxydrhpqa
## 16 53.902     NC vcjmutngdo
## 22  6.030     NC ztnqhxgwfs
```

Now, we unstack:

```
merged2 <- unstack(dmyc0, form = myc ~ cond)
merged2
```

```
##    Cancer     NC
## 1  58.402 59.427
## 2  22.089 23.497
## 3  24.621 24.396
## 4  43.832 44.679
## 5   0.300  1.675
## 6  31.055 32.260
## 7  52.626 54.494
## 8  38.289 39.634
## 9  29.723 30.300
## 10 76.188 78.361
## 11 54.079 53.902
## 12  6.190  6.030
```

We could directly use the data, but we are missing the IDs. We will add them:

```
(merged2$id <- dmyc0$id[1:12])
```

```
##  [1] amcjgskyfu cfxzokirqa dcwzqfvxno gdqxtljvuf idgcwfblvz jtqbflruwv jtrymxigds
##  [8] krupfebolw qwurdtxljs tuxydrhpqa vcjmutngdo ztnqhxgwfs
## 12 Levels: amcjgskyfu cfxzokirqa dcwzqfvxno gdqxtljvuf idgcwfblvz ... ztnqhxgwfs
```

Verify we are now ok comparing merged2 with dmyc.

### 8.2.4 Reshaping with `reshape`

This is another built-in function in R, `reshape`. This can be a complicated function to use, but it is extremely powerful. I'll use it here fully specifying all the arguments[5]:

```
(merged3 <- reshape(dmyc, direction = "wide", idvar = "id",
                    timevar = "cond", v.names = "myc"))
```

```
##            id myc.Cancer myc.NC
## 1  krupfebolw     38.289 39.634
## 2  tuxydrhpqa     76.188 78.361
## 3  dcwzqfvxno     24.621 24.396
## 4  vcjmutngdo     54.079 53.902
## 5  gdqxtljvuf     43.832 44.679
## 6  idgcwfblvz      0.300  1.675
## 7  jtqbflruwv     31.055 32.260
## 8  amcjgskyfu     58.402 59.427
## 9  qwurdtxljs     29.723 30.300
## 10 ztnqhxgwfs      6.190  6.030
## 11 jtrymxigds     52.626 54.494
## 12 cfxzokirqa     22.089 23.497
```

[5] In this case, you do not need to specify the `v.names` argument, for example, but I'd rather be explicit.

### 8.2.5 Reshaping with `spread`

OK, so we will do it in yet another way. You need to install the *tidyr* package. If you don't have it, install it if you want. This is not a big deal: it is just to show you a fourth way.

```
library(tidyr)
(merged4 <- spread(dmyc, cond, myc))
```

```
##            id Cancer     NC
## 1  amcjgskyfu 58.402 59.427
## 2  cfxzokirqa 22.089 23.497
## 3  dcwzqfvxno 24.621 24.396
## 4  gdqxtljvuf 43.832 44.679
## 5  idgcwfblvz  0.300  1.675
## 6  jtqbflruwv 31.055 32.260
## 7  jtrymxigds 52.626 54.494
## 8  krupfebolw 38.289 39.634
## 9  qwurdtxljs 29.723 30.300
## 10 tuxydrhpqa 76.188 78.361
## 11 vcjmutngdo 54.079 53.902
## 12 ztnqhxgwfs  6.190  6.030
```

You can compare with `merged3` and `merged2` and `merged`, and verify they have the same data.

### 8.2.6 `dcast` from *reshape2*?

As it says, you could try doing the reshaping using `dcast` from *reshape2*. Try it if you want to.

### 8.2.7 Reshaping et al: final comments

A few comments:

- This data reshaping is not strictly necessary from doing a paired t-test in R.
- We will see below a way to carry out these same analyses with a linear model (section 8.7). And no reshaping needed.
- Data manipulation can be tricky: you **must** be sure to get it right. What would I do? Do it in two different ways, and compare. Among those two I'd probably have `reshape` because, even if complicated, it allows you to be completely explicit about what is what.

## 8.3 The paired t-test

Just do a paired t-test. What is the result? Compare it with doing a t-test as if they were two independent samples. The differences are rather dramatic!

I'll use `merged3`; you can use `merged2` or `merged4` if you want.

```
## Paired
t.test(merged3$myc.NC, merged3$myc.Cancer, alternative='two.sided',
       conf.level=.95,  paired=TRUE)
```

```
##
##  Paired t-test
##
```

```
## data:  merged3$myc.NC and merged3$myc.Cancer
## t = 4.079, df = 11, p-value = 0.001823
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.432056 1.444777
## sample estimates:
## mean of the differences
##               0.9384167
```

```
## Two-sample
t.test(merged3$myc.NC, merged3$myc.Cancer, alternative='two.sided',
       conf.level=.95,  paired=FALSE)

##
##  Welch Two Sample t-test
##
## data:  merged3$myc.NC and merged3$myc.Cancer
## t = 0.10365, df = 21.996, p-value = 0.9184
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -17.83752  19.71435
## sample estimates:
## mean of x mean of y
##  37.38792  36.44950
```

The last is of course the same as

```
t.test(myc ~ cond, alternative = 'two.sided', conf.level=.95,
       var.equal=FALSE,  data=dmyc)

##
##  Welch Two Sample t-test
##
## data:  myc by cond
## t = -0.10365, df = 21.996, p-value = 0.9184
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -19.71435  17.83752
## sample estimates:
## mean in group Cancer     mean in group NC
##             36.44950             37.38792
```

(Is there a sign difference? Why? What happens if you change the order of NC and Cancer in the paired test?)

## 8.4 The paired t-test again: a single-sample t-test

What is the above test doing?

Create a new variable (e.g., call it "diff.nc.c") that is the difference of myc in the NC and Cancer subjects and do a one-sample t-test.

```
diff.nc.c <- (myc.nc - myc.cancer)
t.test(diff.nc.c)
```

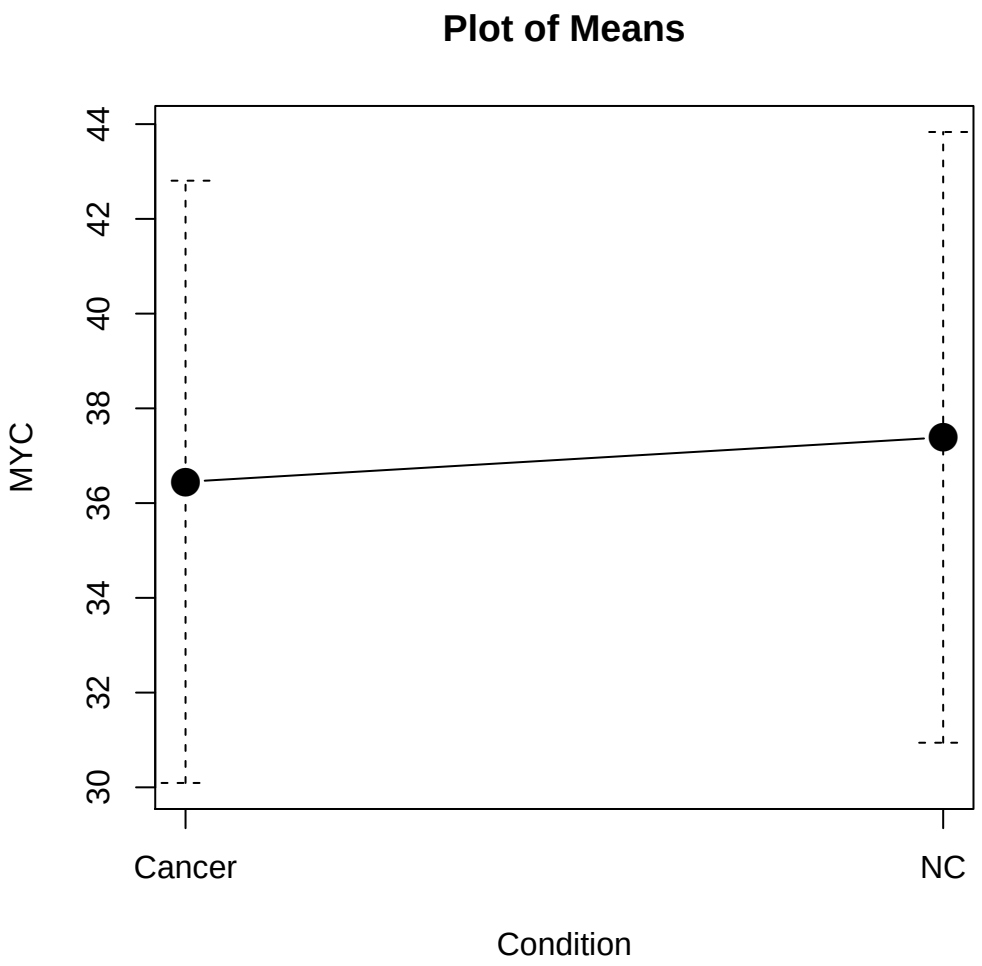
```
##
##  One Sample t-test
##
## data:  diff.nc.c
## t = 4.079, df = 11, p-value = 0.001823
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.432056 1.444777
## sample estimates:
## mean of x
## 0.9384167
```

Compare the t-statistic, the degrees of freedom and the p-value with the paired t-test we did above. Again: what is it we are doing with the paired t-test? (Remember this, as this will be crucial when we use Wilcoxon's test).

## 8.5 Plots for paired data

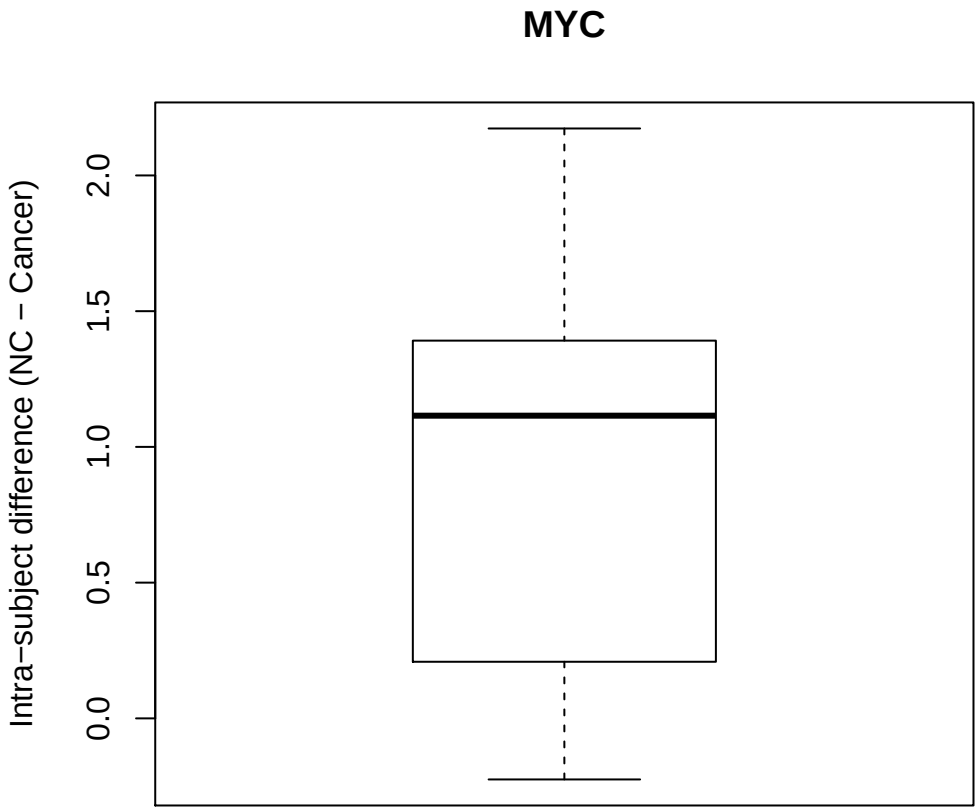
What do you think of this plot?

```
plotMeans(dmyc$myc, dmyc$cond, error.bars = "se", ylab = "MYC",
          xlab = "Condition")
```



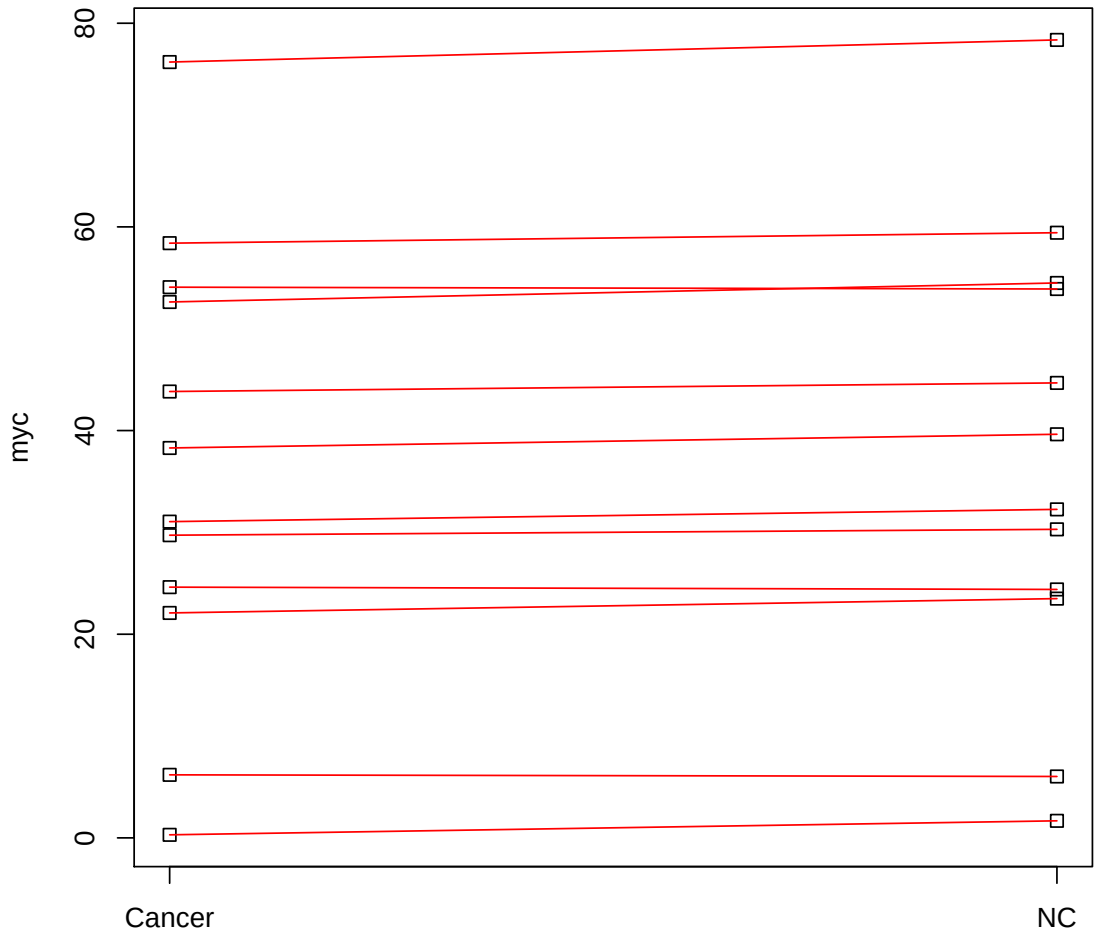

So what is a reasonable plot for paired data? A boxplot (or stripchart if few points) of the within-individual (or Intra-subject) differences is a very good idea.

```
Boxplot( ~ diff.nc.c, data = merged3, xlab = "",
         ylab = "Intra-subject difference (NC - Cancer)", main = "MYC")
```


MYC
Intra−subject difference (NC – Cancer)
0.0
0.5
1.0
1.5
2.0

A more elaborate plot directly shows both the NC and Cancer data, with a segment connecting the two observations of a subject(beware: this code does the job but it is not very efficient or elegant):

```
stripchart(myc ~ cond, vertical = TRUE, data = dmyc)
for(i in unique(dmyc$id))
  segments(x0 = 1, x1 = 2,
           y0 = dmyc$myc[dmyc$cond == "Cancer" & dmyc$id == i],
           y1 = dmyc$myc[dmyc$cond == "NC" & dmyc$id == i],
           col = "red")
```




Alternatively (though I do not see much of a gain here) we could have done:

```
stripchart(myc ~ cond, vertical = TRUE, data = dmyc)
f1 <- function(x) {
    segments(x0 = 1, x1 = 2,
             y0 = dmyc$myc[dmyc$cond == "Cancer" & dmyc$id == x],
             y1 = dmyc$myc[dmyc$cond == "NC" & dmyc$id == x],
             col = "red")
}
sapply(unique(dmyc$id), f1)
```

Now, look at these plots carefully, and understand that there are different sources of variation. In this particular example, there was a huge inter-subject variation in the expression of MYC; if we can control for it (e.g., with intra-subject measures) then we can detect what is, in relative terms, a small effect due to the condition.

(This is a short section, but it is very important. That is the reason I have added a few plots to beef it up. Seriously.)

## 8.6 Choosing between paired and two-sample t-tests

**How the data were collected dictates the type of analysis**. This is a crucial idea. You cannot conduct a two-sample t-test when your data are paired because your data are NOT independent (see also section 12).

This section is not about the analysis, but about the design. It is about "should I collect data so that data are paired or not?" A paired design controls for subject effects (correlation between the two measures of the same subject) but if there are none, then we are decreasing the degrees of freedom (by half): compare the degrees of freedom from the paired and the non-paired tests on the myc data. In most cases with biological data there really are subject effects that lead to large correlations of within-subject measurements. But not always.

This is a major topic (that cannot be done justice to in a paragraph). Sometimes the type of data are something you have no control over. But sometimes you do have control. How do you want to spend your money? Suppose you can sequence 100 exomes. Do you want to do 100 samples, 50 controls and 50 tumors, or do you want to do those same 100 exomes, but from 50 patients, getting tumor and non-tumor tissue? The answer is not always obvious. Go talk to a statistician.

## 8.7 A first taste of linear models

In the paired test, we implicitly have a model like this:

$$Expression.of.MYC = function(subject\ and\ condition) + \epsilon$$

which we make simpler (assuming additive contributions of each factor) as

$$Expression.of.MYC = effect.of.subject + effect.of.condition + \epsilon$$

Lets go and fit that model!

```
LinearModel.1 <- lm(myc ~ id + cond, data = dmyc)
summary(LinearModel.1)
```

```
##
## Call:
## lm(formula = myc ~ id + cond, data = dmyc)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.6173 -0.2224  0.0000  0.2224  0.6173
##
## Coefficients:
##             Estimate Std. Error  t value Pr(>|t|)
```

```
## (Intercept)   58.4453     0.4147  140.918  < 2e-16 ***
## idcfxzokirqa -36.1215     0.5635  -64.098 1.65e-15 ***
## iddcwzqfvxno -34.4060     0.5635  -61.054 2.82e-15 ***
## idgdqxtljvuf -14.6590     0.5635  -26.013 3.14e-11 ***
## ididgcwfblvz -57.9270     0.5635 -102.793  < 2e-16 ***
## idjtqbflruwv -27.2570     0.5635  -48.368 3.62e-14 ***
## idjtrymxigds  -5.3545     0.5635   -9.502 1.23e-06 ***
## idkrupfebolw -19.9530     0.5635  -35.407 1.10e-12 ***
## idqwurdtxljs -28.9030     0.5635  -51.289 1.90e-14 ***
## idtuxydrhpqa  18.3600     0.5635   32.580 2.72e-12 ***
## idvcjmutngdo  -4.9240     0.5635   -8.738 2.80e-06 ***
## idztnqhxgwfs -52.8045     0.5635  -93.703  < 2e-16 ***
## condNC         0.9384     0.2301    4.079  0.00182 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5635 on 11 degrees of freedom
## Multiple R-squared:  0.9997,Adjusted R-squared:  0.9993
## F-statistic:  2840 on 12 and 11 DF,  p-value: < 2.2e-16
```

Run it and look at the line that has "cond" (so, using your common sense, ignore all the "id[blablablabla]" lines). What is the t-value and the p-value for "cond"?

This is a linear model. And this particular one is also a two-way ANOVA.

```
Anova(LinearModel.1, type = "II")
```

```
## Anova Table (Type II tests)
##
## Response: myc
##            Sum Sq Df  F value    Pr(>F)
## id        10815.9 11 3096.230 < 2.2e-16 ***
## cond          5.3  1   16.638  0.001823 **
## Residuals     3.5 11
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Relate this to the figures above. This should all make sense. Regardless, linear models and ANOVAs will be covered in much more detail . . . in the next part!

# 9 One-sample t-test

We have already used the one-sample t-test (section 8.4). You can of course compare the mean against a null value of 0, but you can also compare against any other value that might make sense ("Null hypothesis mu" in R commander). Issues about assumptions are as for the two-sample (except, of course, there are no considerations of differences in variances among groups, since there is only one group). One-sample tests are probably not as common in many cases, as we are often interested in comparing two groups. But when you want to compare one group against a predefined value, the one-sample t-test is what you want.

# 10 Non-parametric procedures

## 10.1 Why and what

Non-parametric procedures refer to methods that do not assume any particular probability distribution for the data. These methods often proceed by replacing the original data by their ranks. They offer some protection against deviations from some assumptions (e.g., deviations from normality). But some times they might be testing a slightly different null hypothesis. Whether or not to use them is a contentious issue. Some considerations that come into play are:

- Do the data look bad enough that a parametric procedure will lead to trouble?
- What about the relative efficiency of the non parametric procedure? (relative efficiency refers to the sample size required for two different procedures that have similar type I error rate to achieve the same power —more on power later). When assumptions are met, parametric methods do have more power (though not always a lot more). When assumptions are not met, nonparametric methods might have more power, or the correct Type I error, etc. Note that very, very, very small p-values are often not achievable with non-parametric methods (because of the way they work), and this is a concern with multiple testing adjustments in omics experiments (this will be covered in the next lesson).
- Is this test flexible enough? In particular, can I accommodate additional experimental factors?

We focus here on the Wilcoxon test. This is often the way to go when we have **ordinal** scale measurements and we want to compare two independent groups. Note, however, that the Wilcoxon test **requires interval scale data** for the single-sample Wilcoxon and the paired Wilcoxon (this is something that many websites and some textbooks get wrong); a simple illustration is shown in section 10.4.

However, **the independence assumption** is as important with Wilcoxon as with the t-test. Nonparametric tests are not "assumption-free" tests!!! (there is no such thing, in statistics or in life).

An excellent non-parametrics book is Conover's "Practical nonparametric statistics", in its 3rd edition (it is so awesome, I have two copies, one at home and one at the office —you can get one from Amazon for about 20 euros). (For more general categorical data analysis, Agresti's "Categorical data analysis" is a must. It is now also in its third edition. There is, by the same author, a smaller "Introduction to categorical data analysis").

## 10.2 Two-sample Wilcoxon or Mann-Whitney test

This test applies to data of ordinal and interval scales. The basic logic is: put all the observations together, rank them, and examine if the sum of the ranks of one of the groups is larger (or smaller) than the sum of the ranks from the other. If this makes sense to you, you should understand why you can use both ordinal and interval scales.

For example:

```
wilcox.test(p53 ~ cond, alternative = "two.sided", data = dp53)
```

```
##
##  Wilcoxon rank sum test
##
## data:  p53 by cond
## W = 22, p-value = 0.006473
```

```
## alternative hypothesis: true location shift is not equal to 0
```

## 10.3 Wilcoxon matched-pairs test

The idea here is to assess whether the within-pair differences are symmetrically distributed around zero. As we are talking about symmetry, this means that this test can be used **only with interval scale** data (see also section 10.4).

In a nutshell, this is what the test does: for each pair, it computes the difference of the two measures (thus, taking differences must make sense, and it does not for ordinal data but it does for interval data). These differences (discarding the sign) are then ranked, and then the sum of the ranks of all the positive differences is computed and compared to the null distribution.

```
wilcox.test(myc.nc, myc.cancer, alternative = 'two.sided', paired = TRUE)
```

```
##
##  Wilcoxon signed rank test
##
## data:  myc.nc and myc.cancer
## V = 72, p-value = 0.006836
## alternative hypothesis: true location shift is not equal to 0
```

You can verify that the results are the same as doing a single sample Wilcoxon, or a Wilcoxon signed-rank test, on the within-subject differences.

```
wilcox.test(diff.nc.c)
```

```
##
##  Wilcoxon signed rank test
##
## data:  diff.nc.c
## V = 72, p-value = 0.006836
## alternative hypothesis: true location is not equal to 0
```

## 10.4 Wilcoxon's paired test and interval data

If you understand the logic of what the two Wilcoxon tests are doing, you should understand why the one for the two-sample case works with ordinal data but the one for the matched pairs requires interval data. This explores the issue further.

In particular, if we were to transform the data using a monotonic non-linear transformation (e.g., a log transformation), the Wilcoxon test for two groups should never be affected, but the one for the paired case could be affected. Why? Because in the first case we rank the values, and the ranks of the values are the same as the ranks of the values after any monotonic transformation. However, the rank of the within-subject differences might differ if we use a transformation on the original data and then rank the within-subject differences.

```
## Without logs
wilcox.test(dmyc$myc[1:12], dmyc$myc[13:24], paired = TRUE)
```

```
##
##  Wilcoxon signed rank test
##
## data:  dmyc$myc[1:12] and dmyc$myc[13:24]
```

```
## V = 6, p-value = 0.006836
## alternative hypothesis: true location shift is not equal to 0

## After taking logs
wilcox.test(log(dmyc$myc[1:12]), log(dmyc$myc[13:24]), paired = TRUE)

##
##  Wilcoxon signed rank test
##
## data:  log(dmyc$myc[1:12]) and log(dmyc$myc[13:24])
## V = 9, p-value = 0.01611
## alternative hypothesis: true location shift is not equal to 0
```

Notice how the statistic and the p-value change. That would not happen if the test could be applied to ordinal data.

It is also easy to create examples that show that the one-sample Wilcoxon does require interval data. This is left as a simple exercise.

### 10.5 A bad way to choose between nonparametric and parametric procedures

Don't do the following:

1. Carry out a test for normality
2. If failed, use a nonparametric test. Otherwise a t-test.

There are several problems with the above procedure, among them:

- Tests for normality do not have a power of 1, and in fact can have very low power with small sample sizes.
- They can have power to detect a minor and irrelevant deviation from normality (e.g., slightly heavy tails) that have no effects on, say, a t-test. So we should really prefer a t-test, but we might be mislead into doing a Wilcoxon.
- They might fail to detect a large deviation from normality in a very non-normal small sample where we really, given our ignorance, might want to conduct a Wilcoxon test.
- They lead to "sequential testing" problems, where the second p-value (the one from the t or Wilcoxon) cannot be simply interpreted at face value (as it is conditional on the normality test).

## 11 Power of a test

How likely we are to detect a difference that really exists (power) depends on:

- The threshold ($\alpha$ level, or Type I error).
- The sample size.
- The size of the effect (difference in means).
- The standard deviation.

Do you understand each one of these? We will not get into details. If you want, install *Rcmdr* and play with the "Teaching demos: Power of a test" (from the *RcmdrPlugin.TeachingDemos*).

There are simple, standard ways of figuring out the power. But they require you to specify the above values. Not always known (or easy to guess). The *IPSUR* and *RcmdrPlugin.IPSUR* packages provide some extra tools.

Power can be computed before hand to:

- Know if we are likely to find a difference if there is one (given our sample size and estimated effect sizes and standard deviations).
- Figure out if our sample size is OK given the desired power (and estimated effect sizes and standard deviations).

Please note: **it makes little sense** to compute the power of a test after the fact. It tells you nothing valuable[6]

# 12 Non-independent data

## 12.1 Multiple measures per subject

Paired data are non-independent: they are associated via the subject, or id. There are other forms of non-dependency. Many. We will now look at the most common one: multiple measures per subject.

We will use the `BRCA2.txt` data set. Import it, etc. Look at the data carefully. It looks like there are 3 observations per subject in a total of 13 subjects. Basically, each subject has been measured 3 times. Thus, there are not 39 independent measures. Again, ask yourself: what is the true "experimental unit" (or observational unit)? It is arguably the subject, which is the one that has, or has not, cancer in this case. The 3 measures per subject are just that: multiple measures of the same experimental unit. This is an idea that should be crystal clear; make sure you understand that.

We can do a t-test ignoring this fact, but it would be wrong. Look at the degrees of freedom:

```
t.test(brca2 ~ cond, data = dbrca)

##
##  Welch Two Sample t-test
##
## data:  brca2 by cond
## t = -2.1969, df = 28.061, p-value = 0.03645
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -3.9309162 -0.1377338
## sample estimates:
## mean in group Cancer     mean in group NC
##             5.953208             7.987533
```

What can we do? The most elegant approaches are not something we can cover here[7]. However, fortunately for us in **this particular case**, all subjects have been measured the same number of times. Thus, we can simply take the mean per subject, and then do a t-test on those mean values per subject.

We want to aggregate (using the mean) the values of brca2, and we want to aggregate by "id". However, we want to keep "cond" also. So we will aggregate by "cond" and "id". In fact, this is a double check that each subject is in one, and only one, of the groups. Call this data "aggbrca".

[6] The paper by Hoenig and Heisey, "The abuse of power: the pervasive fallacy of power calculations for data analysis", *The American Statistician*, 2001, 55: 19–24, explains this in more detail.

[7] A general approach is a mixed-effects linear model with random effects for subjects; alternatively, and in this simple case, we can use an ANOVA with the correct error term, for instance from a multistratum linear model similar to the ones used to analyze split-plot experiments.

```
aggbrca <- aggregate(dbrca[,c("brca2"), drop = FALSE],
                     by = list(cond = dbrca$cond,
                         id = dbrca$id), FUN = mean)
```

You can also call aggregate in a simpler way in this case:

```
aggbrca <- aggregate(brca2 ~ cond + id, data = dbrca, FUN = mean)
```

We want to double check that we have the exact same number of measures per subject. How? A simple procedure is to aggregate again (give the output another name) but using "length" instead of "mean". Now we will be aggregating over only "id" (so we aggregate brca2 and cond) and also over both "id" and "cond", to double check. Do you understand why we are doing this?

```
aggregate(brca2 ~ cond + id, data = dbrca, FUN = length)
```

```
##      cond         id brca2
## 1  Cancer exsapkiqfh     3
## 2  Cancer ibrwvgtenp     3
## 3  Cancer jrpugeqniv     3
## 4  Cancer kuxsftbalz     3
## 5      NC lmnksqwajv     3
## 6  Cancer lsgbpqhdnv     3
## 7      NC ltfcvdrqnu     3
## 8      NC ptqoiuvgxe     3
## 9  Cancer udcmvzqrln     3
## 10 Cancer wfaebcghyn     3
## 11 Cancer wypasvcomk     3
## 12     NC xmsigtnyrl     3
## 13     NC xtljfdovgh     3
```

```
aggregate(brca2 ~ id, data = dbrca, FUN = length)
```

```
##            id brca2
## 1  exsapkiqfh     3
## 2  ibrwvgtenp     3
## 3  jrpugeqniv     3
## 4  kuxsftbalz     3
## 5  lmnksqwajv     3
## 6  lsgbpqhdnv     3
## 7  ltfcvdrqnu     3
## 8  ptqoiuvgxe     3
## 9  udcmvzqrln     3
## 10 wfaebcghyn     3
## 11 wypasvcomk     3
## 12 xmsigtnyrl     3
## 13 xtljfdovgh     3
```

```
aggregate(. ~ id, data = dbrca, FUN = length)
```

```
##            id brca2 cond
## 1  exsapkiqfh     3    3
## 2  ibrwvgtenp     3    3
## 3  jrpugeqniv     3    3
## 4  kuxsftbalz     3    3
## 5  lmnksqwajv     3    3
## 6  lsgbpqhdnv     3    3
```

```
## 7  ltfcvdrqnu     3    3
## 8  ptqoiuvgxe     3    3
## 9  udcmvzqrln     3    3
## 10 wfaebcghyn     3    3
## 11 wypasvcomk     3    3
## 12 xmsigtnyrl     3    3
## 13 xtljfdovgh     3    3
```

Now, let's do a t-test on the aggregated data. Pay attention to the degrees of freedom (and the statistic and p-values):

```
t.test(brca2 ~ cond, alternative = 'two.sided', conf.level = .95, var.equal = FALSE,
  data = aggbrca)
```

```
##
##  Welch Two Sample t-test
##
## data:  brca2 by cond
## t = -1.2832, df = 8.0293, p-value = 0.2352
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -5.687861  1.619211
## sample estimates:
## mean in group Cancer     mean in group NC
##             5.953208             7.987533
```

You can see that doing things correctly makes, in this case, a large difference. When we do things incorrectly, we can end up believing that there is a strong effect when, really, there is no evidence of effect.

What if the subjects had been measured a different number of times? What would be the problems of simply taking the averages over subjects?

## 12.2 Nested or hierarchical sources of variation

A general way of thinking about these issues is that we can have nested, or hierarchical, levels of variation (e.g., multiple measures per cell, multiple cells per subject, multiple subjects for two different treatments) and it is crucial to understand what each level of variation is measuring (e.g., the difference between technical and biological variation) and to conduct the statistical analysis in a way that do incorporate this nestedness. A very recent introductory 2-page paper with biomedical applications is Blainey et al., 2014, "Points of significance: Replication", in *Nature Methods*, 2014, 11, 879–880, where they discuss issues of sample size choice at different levels.

By the way, not surprisingly, split-plot ANOVA and nested ANOVA are typical, traditional, ways of analyzing these kinds of designs. Many of these issues are, thus, naturally addressed in the context of linear models.

## 12.3 Non independent data: extreme cases

In the extreme, this problem can lead to data that should not be analyzed at all because there is, plainly, no statistical analysis that can be carried out. For instance, suppose we want to examine the hypothesis that consuming DHA during pregnancy leads to increased mielinization in the

hippocampus of the progeny. An experiment is set to test this idea, comparing the effects on mielinization of mice that have been born to female mice with and without a DHA supplement[8].

Now, a colleague comes to you with the data. She claims there are 40 data points, 20 from brains of newborn mice under the DHA supplement and 20 from brains of newborn mice without the supplement. OK, it looks good: it seems we can carry out an independent samples t-test with about 38 degrees of freedom. However, on asking questions, this is what you find out:

- A total of two female mice were used. One female was given food with DHA and one female without. They were fed the specified diet, and then they mated and got pregnant.
- From the litter for each female, five newborn mice were sacrificed immediately after birth, and four tissue slides were prepared from each brain (thus, 20 data points per female).

No statistical analysis can be conducted here to examine the effects of DHA: there is only one data point per experimental unit. We cannot do a t-test in the right way: there are no degrees of freedom because there is no way to estimate the variance. To put it simply, there is no way to tell whether any differences that could be seen are due to DHA or to the female herself or to any other factor that might have been confounded with the single female per treatment (color of the gloves of the technician or side in the animal room or how nice the male was while mating or ...).

It is this simple: **nothing** can be said from this data about the effects of DHA. (It is not even worth importing the data for this analysis. Maybe it is interesting to get a preliminary idea of the intra-brain variation in mielinization, but not for the original question of the DHA effect).

There are some recent papers about issues like this that go over the pseudoreplication idea (e.g., Lazic, 2010, *BMC Neuroscience*) and this is probably a pervasive (but difficult to detect) problem. This kind of meaningless analysis can lead to lots of non-reproducible results.

Which brings us to the fundamental idea of **thinking carefully about the experimental design**, something we will take a quick look at in the linear models section.

## 12.4 More non-independences and other types of data

What if some subjects had cousins and brothers in the data set? And if some of them came from the same hospital and other from other hospitals? And ... ? This all lead to multilevel and possible crossed terms of variance. Mixed effects models can be used here. Go talk to a statistician (after looking at the rest of these notes).

However, for simple cases and to get going while we talk to the statistician, the approach we used above (collapsing the data over lower levels, leaving data that are independent at the experimental unit level) can some times be used in other scenarios. The independence assumption is actually crucial in many statistical analysis, be they t-tests, ANOVAs, chi-squares, regressions, etc, etc. (As has been mentioned repeatedly, there are ways to incorporate or deal with the non-independence, for categorical, ordinal, and interval data, but they are far from trivial).

To make the point more clear, this is an example from the data for the TFM ("trabajo fin de master") from a former student of BM-1[9]. Briefly, she was interested in the rates of chromosomal aberrations in different types of couples that went to a fertility clinic. For instance, suppose you want to examine incidence of aneuploidy in embryos from two groups of fathers, "younger fathers" and "older fathers". The simplest idea here is to use a chi-square ($\mathcal{X}^2$) test to compare the frequency of aneuploidies between older and younger fathers (you will see chi-square tests in

[8] This is based on an actual data set I was once asked to analyze. I've changed enough details —no DHA or anything similar. But the outcome was the same: no analyses were possible at all.

[9] This is true. I am not making it up.

Lesson 4). But the problem is that each couple (each father in this case) contributes multiple embryos and we cannot simply do a chi-square counting embryos, as we would again run into a non-independence problem. A simple approach, especially if each father contributes the same number of embryos, is to calculate, for each father, the proportion of embryos with aneuploidies. And then, to examine if that per-father proportion of aneuplodies differs between older and younger fathers with, say, an independent samples t-test (possibly of suitably transformed data) or a two-samples Wilcoxon test.

*Part II*

# *Linear models: ANOVA, regression, ANCOVA*

## 13 Introduction to ANOVAs, regression, and linear models

Linear models and their extensions (which include logistic regression, but also survival analysis, many classification problems, non-linear models, analysis of experiments, dealing with many types of dependent data, etc, etc) are one fundamental topic in statistics. Here, we will only scratch the surface. But you should come away from this lesson understanding that these methods are extremely powerful and flexible and that they can be used to address a huge variety of different research questions. You would spend your time wisely if you at least took a look at some of the references we provide at the end. To emphasize again: ANOVAs, regression, ANCOVAs, are just special type of linear models; the terminology is not that important, but I'll use those terms as they might be more familiar to you.

### 13.1 Files we will use

- This one
- `MIT.txt`
- `Cholesterol.txt`
- `AnAge_birds_reptiles.txt`
- `CystFibr2.txt`

## 14 Comparing more than two groups

### 14.1 Recoding variables

Import `MIT.txt` and call the object `dmit`. These are data about mitochondrial activity related to three different training regimes.

```
dmit <- read.table("MIT.txt", header = TRUE)
```

Whoever entered the data, however, used a number for "training", which is misleading, because this is really a categorical variable. The first thing we must do, then, is fix that.

Note the `factor` function call:

```
dmit$ftraining <- factor(dmit$training,
                         labels = c('Morning','Lunch','Afternoon'))
```

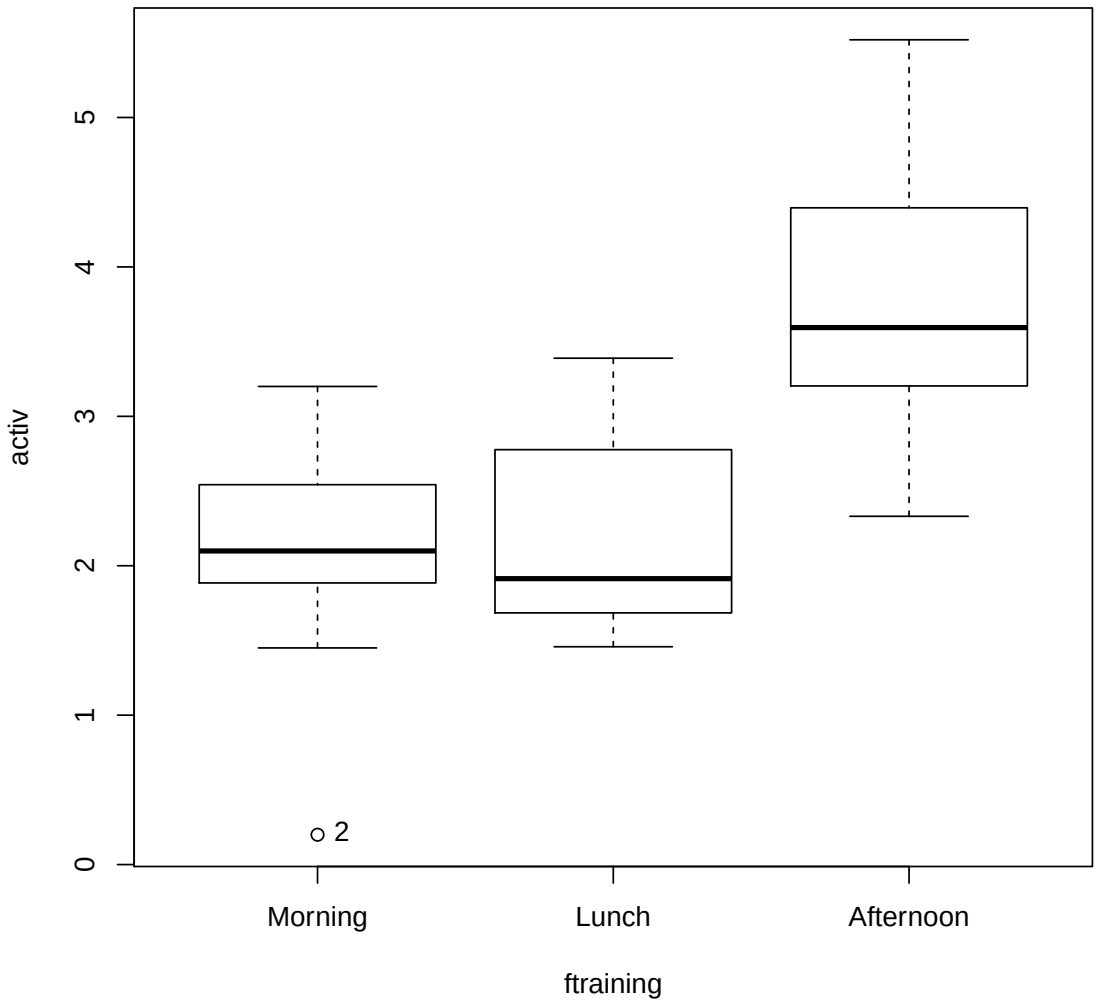
As usual, make sure to look at the data set.

If we were to not recode the factor (or use the original "training") it would be a disaster (look at the output in section A).

## 14.2 A boxplot

You are advised to also plot the data. For instance, this will do:

```
Boxplot(activ ~ ftraining, data = dmit, id.method = "y")
```




(The output might show a "2"; that is the identifier —row name— of a point that has been flagged as a potential outlier and we will silent that output from now on in these notes).

## 14.3 An ANOVA

We want to see if time of exercise makes any difference. Conducting three t-tests is not the best way to go here: our global null hypothesis is $\mu_{Morning} = \mu_{Lunch} = \mu_{Afternoon}$ and that is what ANOVA will allow us to test directly.

```
AnovaMIT <- aov(activ ~ ftraining, data = dmit)
summary(AnovaMIT)
```

Can you interpret the output?

We will cover this in more detail in class, in case you do not remember ANOVA. Things to notice:

- The two rows; one of them is the effect you are interested in (ftraining)
- The "Df" column: those are the degrees of freedom (three groups - 1 for "ftraining").
- The two columns Sum Sq (Sum of Squares) and Mean Sq (Mean Squares). Sum of Squares is a quantity related to the variance. Mean Squares is obtained from the ratio of Sum Sq over Df. Then, we use Mean Sq to compare how much variance there is between groups related to the variance within groups: the F value is the ratio of Mean Sq of ftraining over Mean Sq of the residuals. The larger that F value, the more evidence there is of groups being different.
- There is a p-value associated with that F value. In this case it is very small.

By the way, notice how we created a "Model" which is called `AnovaMIT`. But we could have named it differently.

## 14.4 So which means are different? Multiple comparisons

That small p-value leads us to reject the null hypothesis $\mu_{Morning} = \mu_{Lunch} = \mu_{Afternoon}$. So there is strong evidence that all three means are not equal. But which one(s) is(are) different from the other(s)?

Let us get some summary data. I will do it in two different ways. This is a convenient one, using a function from *RcmdrMisc*:

```
numSummary(dmit$activ , groups = dmit$ftraining, statistics = c("mean", "sd"))
```

```
##               mean        sd data:n
## Morning   2.103000 0.8113702     11
## Lunch     2.198833 0.6608995     12
## Afternoon 3.797348 0.9013205     23
```

But of course, you can get a similar thing building what you want from scratch with, for instance, `aggregate`

```
with(dmit, aggregate(activ, list(Training = ftraining),
                     function(x) c(mean = mean(x),
                                   sd = sd(x),
                                   n = sum(!is.na(x)))
                     ))
```

```
##    Training     x.mean       x.sd        x.n
## 1   Morning  2.1030000  0.8113702 11.0000000
## 2     Lunch  2.1988333  0.6608995 12.0000000
## 3 Afternoon  3.7973478  0.9013205 23.0000000
```

or by:

```
with(dmit, by(activ, ftraining,
              function(x) c(mean = mean(x),
                            sd = sd(x),
                            n = sum(!is.na(x)))
              ))
```

```
## ftraining: Morning
##       mean         sd          n
##  2.1030000  0.8113702 11.0000000
## ------------------------------------------------------------
## ftraining: Lunch
##       mean         sd          n
##  2.1988333  0.6608995 12.0000000
## ------------------------------------------------------------
## ftraining: Afternoon
##       mean         sd          n
##  3.7973478  0.9013205 23.0000000
```

You can get an idea: it seems that Morning and Lunch are very similar to each other, but Afternoon is very different. This agrees with the impression we got from the boxplot. But we would like a more formal procedure: we are going to compare all pairs of means and we will take into account that we are carrying out multiple comparisons (tests of pairs of means when the ANOVA is not significant are rarely justified[10].

**Comparing all pairs of means** is done using the ANOVA model, so the results are not identical to comparing using t-tests (briefly: the estimate of the variance might be slightly different, and probably better).

**Multiple testing corrections** are needed because we are now conducting three separate tests (in general, if there are $K$ groups and if you compare all pairs of means you carry out $\binom{K}{2} = \frac{K(K-1)}{2}$ tests). Here, we will control the family-wise error rate, the probability of falsely rejecting one or more tests over the family of tests performed —three in our case. The logic is somewhat like that of being struck by lightning: the chances of it happening are extremely small, but every year people die from lightning because the probability that at least one person is killed is huge since we have lots of people exposed to the risk. So even if all null hypotheses for our three tests are true:

- $\mu_{Morning} = \mu_{Lunch}$
- $\mu_{Morning} = \mu_{Afternoon}$
- $\mu_{Lunch} = \mu_{Afternoon}$

if we run the three comparisons, the chances of incorrectly rejecting at least one of the null hypotheses is larger than, say, 0.05 if we simply look at each one of the three p-values and keep any with a p-value $\leq 0.05$. You will see more about multiple testing below (15).

So we will get both a plot and textual output[11]

```
library(multcomp) ## for glht
```

```
## Loading required package: mvtnorm
```

```
## Loading required package: survival
```

[10] Unless some specific, small number of, tests of specific pairs had been planned before the experiment.

[11] If you have used R Commander, you will see a lot of resemblance with the output produced by R Commander. I confess I cheated here; I got the commands below from R Commander, which I found very simple to do by fitting an ANOVA in the menu and then making sure to click on "Pairwise comparisons of means". However, that is just a detail that explains the names of the objects. The procedure using `glht` does not depend on R Commander.


**Figure 1** – Plot of pairwise differences with Tukey contrasts

```
## Loading required package:  TH.data

## Loading required package:  MASS

##
## Attaching package:  'TH.data'

## The following object is masked from 'package:MASS':
##
##    geyser

## The next two lines carry out the multiple comparisons and the following
## lines plot them
Pairs <- glht(AnovaMIT, linfct = mcp(ftraining = "Tukey"))
summary(Pairs) # pairwise tests
confint(Pairs) # confidence intervals
cld(Pairs) # compact letter display
old.oma <- par(oma = c(0,5,0,0))
plot(confint(Pairs))
par(old.oma) ## restore graphics windows settings
```

Look carefully at the plot in Figure 1: for each difference (for each **contrast**), it shows the estimate and a 95% confidence interval around it. The plot title says "95% family-wise confidence interval", and that indicates that multiple testing correction has been used. (You might want to

make that even more explicit by using a title such as "95% family-wise confidence interval using Tukey contrasts").

Given how far two of the contrasts are from 0.0, it seems those are highly significant differences.

The numerical output explicitly shows that we are using Tukey's method and it shows the p-values of each contrast (each comparison), and it makes it clear that we are being reported adjusted p values. There is strong evidence of a difference between Afternoon and the other two levels, but no evidence of differences between Lunch and Morning.

### 14.4.1 And can I plot the means with s.e from the model?

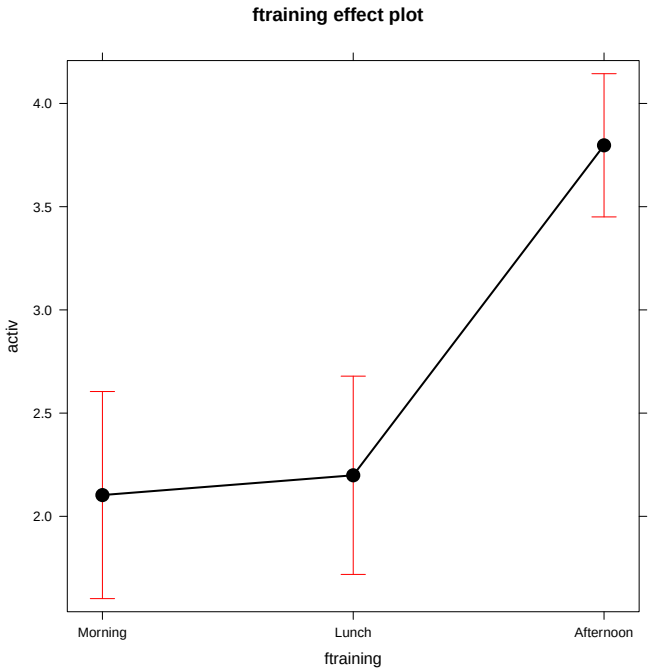
Sure. A simple way of doing it is using the `allEffects` function from the *effects*:

```
library(effects)

##
## Attaching package:  'effects'

## The following object is masked from 'package:car':
##
##     Prestige

plot(allEffects(AnovaMIT), ask = FALSE)
```



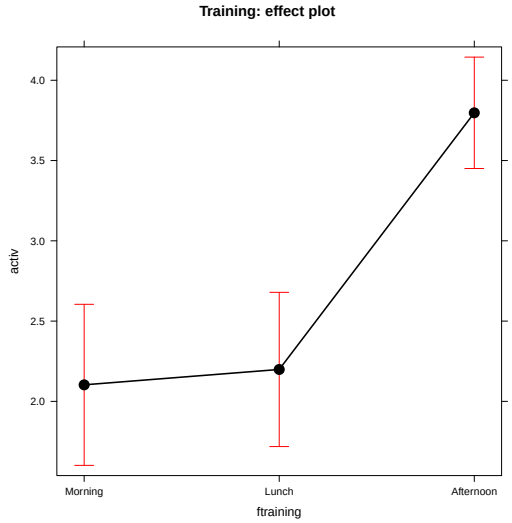

That shows the estimates for each group and a 95% confidence interval (again, based on the whole ANOVA model). But from that figure it is not easy to tell which pairs differ, especially taking multiple comparisons into account.

### 14.4.2 This is a mess. What figures do I use?

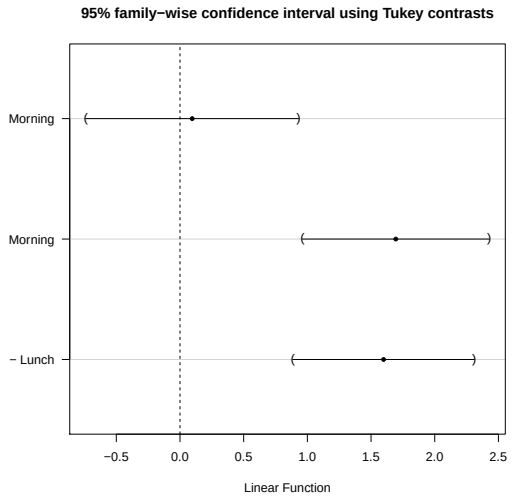
That is up to you :-). But this can work: present both the original means and the plot with the contrasts.

I would actually modify slightly the title of both figures, so that they look better:

```
plot(allEffects(AnovaMIT), ask=FALSE, main = "Training: effect plot")
```




```
.Pairs <- glht(AnovaMIT, linfct = mcp(ftraining = "Tukey"))
tmp <- cld(.Pairs) ## silent assignment
plot(confint(.Pairs),
     main = "95% family-wise confidence interval using Tukey contrasts")
```




### 14.4.3 Side note: Interpreting confidence intervals

If this is not obvious to you, ask it in class: a figure that shows an estimate (e.g., a mean) and a 95% confidence interval, where the interval goes from, say, 1 to 2, **should not** be interpreted as saying that there is a 95% probability that the mean is between 1 and 2. That is not the correct interpretation of a confidence interval. Make sure you understand this!!!

### 14.4.4 Multiple comparisons, other contrasts, etc

There is a wide literature on methods for adjusting for multiple comparisons in ANOVA and linear models. And sometimes a distinction is made between pre-planned and post-hoc comparisons. Tukey's approach is a widely accepted one (though there are others) and the distinction between pre-planned and post-hoc does not arise when researchers directly want to do all possible pairs

right from the beginning. However, many of these issues can become important if you know, from the start, that some comparisons do not matter to you, and/or there are many groups. As well, we could be interested in other types of contrasts, for instance, that the mean of groups 2 and 3 is different from the mean of group 4. Etc, etc. We will not get into this.

### 14.4.5 t-test as ANOVA

Of course, in general, you can just carry out any two-group comparison as an ANOVA. There is nothing wrong with that (and there is a simple correspondence between a t statistic and an F statistic).

### 14.4.6 Several ways of obtaining summaries

For example with `aggregate`

```
with(dmit, aggregate(activ, list(Training = ftraining),
                     function(x) c(mean = mean(x),
                                   sd = sd(x),
                                   n = sum(!is.na(x)))
                     ))
```

```
##    Training     x.mean       x.sd        x.n
## 1   Morning  2.1030000  0.8113702 11.0000000
## 2     Lunch  2.1988333  0.6608995 12.0000000
## 3 Afternoon  3.7973478  0.9013205 23.0000000
```

or by:

```
with(dmit, by(activ, ftraining,
              function(x) c(mean = mean(x),
                            sd = sd(x),
                            n = sum(!is.na(x)))
              ))
```

```
## ftraining: Morning
##       mean         sd          n
##  2.1030000  0.8113702 11.0000000
## ------------------------------------------------------------------------
## ftraining: Lunch
##       mean         sd          n
##  2.1988333  0.6608995 12.0000000
## ------------------------------------------------------------------------
## ftraining: Afternoon
##       mean         sd          n
##  3.7973478  0.9013205 23.0000000
```

### 14.4.7 Can you do an ANOVA with only one sample per group?

Why or why not?

What happens here?

```
y <- c(1, 2, 3)
gr <- factor(c("g1", "g2", "g3"))
anova(lm(y ~ gr))
```

```
## Warning in anova.lm(lm(y ~ gr)): ANOVA F-tests on an essentially perfect fit
are unreliable
```

```
## Analysis of Variance Table
##
## Response: y
##           Df Sum Sq Mean Sq F value Pr(>F)
## gr         2      2       1
## Residuals  0      0
```

```
summary(aov(y ~ gr))
```

```
##             Df Sum Sq Mean Sq
## gr           2      2       1
```

And here?

```
y2 <- c(1, 2, 3, 4)
gr2 <- factor(c("g1", "g2", "g3", "g3"))
anova(lm(y2 ~ gr2))
```

```
## Analysis of Variance Table
##
## Response: y2
##           Df Sum Sq Mean Sq F value Pr(>F)
## gr2        2    4.5    2.25     4.5 0.3162
## Residuals  1    0.5    0.50
```

```
summary(aov(lm(y2 ~ gr2)))
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## gr2          2    4.5    2.25     4.5  0.316
## Residuals    1    0.5    0.50
```

## 14.5 One way ANOVA: summary of steps

1. Enter the data.
2. Recode the factor (the independent variable), if needed.
3. Run the model.
4. Assess model diagnostics (see section 23).
5. Carry out comparisons between pairs of means with appropriate adjustment for multiple comparisons.

| | Null hypothesis not rejected | Null hypothesis rejected |
|---|---|---|
| Means do not differ ($H_0$ *true*) | U | V |
| Means differ ($H_0$ *false*) | T | S |

**Table 3** – Multiple comparisons. In rows is the "truth" (how things really are), and in columns the output from our testing procedure (what we end up claiming or believing). The sum of all entries is the total number of comparisons made.

# 15 Multiple comparisons: FWER and FDR

## 15.1 Family-wise error rate

In section 14.4 we covered multiple comparisons. Here we go into a little bit more detail before continuing with ANOVA. As in section 14.4, suppose we are testing a number of null hypothesis. Table 3 shows a depiction of what we are concerned about, where the letters in each cell refer to the number of means tested that fall in each case.

In the example in section 14.4 $U + V + T + S = 3$ (beware, 3 is the number of **hipothesis tests**, it is not the number of means; in our case, both are three, but Table 3 reflects number of hypothesis, not tests). Procedures such as the one we used (Tukey) or Bonferroni or similar ones, try to control the probability that $V \geq 1$. They control what is called the "family-wise error rate" (FWER).

The intuitive idea is: "I want to control very tightly the probability of falsely rejecting any hypothesis", and that is the same as saying "I want to control very tightly the probability that $V$ is equal or larger than 1". (I use the expression "control very tightly" because if we insist in "I NEVER want to falsely reject any null hypothesis" then ... we will never reject any null hypothesis). What Tukey, Bonferroni, and other procedures for controlling the family wise error rate provide are mechanisms for ensuring that $Pr(V \geq 1)$ is below a number you specify (e.g., 0.05).

Note that, in our usage of Tukey, we did not pre-specify that $Pr(V \geq 1)$. The procedure is run, and it gives us "adjusted p-values". And what is an adjusted p-value? The classical paper from Wright, 1992, "Adjusted p-values for simultaneous inference", *Biometrics*, 48: 1005–1013, has in p. 1006 this definition of adjusted p-value: "The adjusted $P$-value for a particular hypothesis within a collection of hypotheses, then, is the smallest overall (i.e., 'experimentwise') significance level at which the particular hypothesis would be rejected." This might sound like a mess, but it really ain't. Think about it. It is so nice that it makes comparisons very simple, as Wright explains: "An adjusted $P$-value can be compared directly with any chosen significance level $\alpha$: If the adjusted $P$-value is less than or equal to $\alpha$, the hypothesis is rejected."

## 15.2 False discovery rate (FDR)

There is a different approach to the multiple testing problem. In this approach we focus on controlling the fraction of false positives. The total number of null hypothesis we reject is $V + S$. The intuitive idea behind the control of the false discovery rate (**FDR**) is to bound (to set an upper limit to) to the ratio $\frac{V}{V+S}$[12].

One key difference is that the FDR can be kept reasonably low (say, 0.01) even when it is almost sure that $V \geq 1$. When could this happen? For instance, when we are conducting tens of thousands of hypothesis tests. Again, the FDR will control the fraction of false discoveries

[12]There are several different approaches. The most common one is to control $FDR = E(Q)$ where $Q = V/(V + S)$ if $V + S > 0$ (and $Q = 0$ otherwise). But there are others, such as the $pFDR$, etc.

whereas the control of the family wise error rate (FWER) is emphasizing that $V$ don't become 1 or more.

As we did with Tukey and the FWER procedures, we generally do not pre-specify the level of FDR we want to attain but, rather, we obtain "adjusted p-values". The difference in the meaning of "adjusted" is that now these p-values are adjusted for FDR (not adjusted for control of the family wise error rate). So, when we deal with FDR, the adjusted p-value of an individual hypothesis is the lowest level of FDR for which the hypothesis is first included in the set of rejected hypotheses (e.g., Reiner et al., 2003, *Bioinformatics*).

The FDR is usually employed in screening procedures, where we are willing to allow some false discoveries, because we are screening over thousands of hypothesis. The cost of requiring $V = 0$ would be to miss many discoveries. One example? Suppose that you have measured the expression of 20000 genes in two sets of subjects some with colon cancer and some without. Now, you can do the equivalent of 20000 t-tests. So you will get 20000 p-values, and you will want to adjust those 20000 tests for multiple testing.

How do you adjust for multiple testing in R? This is easily done with the function `p.adjust`. When you are applying FDR you often have a collection of p-values already.

I will make a simple example up and will only use four p-values (not 20000) for the sake of simplicity. Suppose we have done a screening procedure, testing four genes. You get the p-values I show below. To use an FDR correction method I use `p.adjust` with the `method = "BH"` argument (BH is one of several possible types of FDR correction). To show what happens, I have then combined the two, side by side, so you can see the original p-value and the FDR-adjusted one.

```
p.values <- c(0.001, 0.01, 0.03, 0.05)
adjusted.p.values <- p.adjust(p.values, method = "BH")
cbind(p.values, adjusted.p.values)
```

```
##      p.values adjusted.p.values
## [1,]    0.001             0.004
## [2,]    0.010             0.020
## [3,]    0.030             0.040
## [4,]    0.050             0.050
```

How do we interpret this? Here I will only cover the very basics. But go back a couple of paragraphs, and re-read the definition of adjusted p-value for the FDR. So, for example, if we keep as "significant" all the genes with a p-value (not adjusted p-value, but p-value, so the last first three) $\leq 0.030$, the FDR (the expected number of false discoveries) will be 0.040 (the FDR-adjusted p-value for the gene with $p-$value of 0.03).

Note that the FDR applies not just to comparisons between means or t-tests, but to any kind of test (comparing variances, correlations, etc).

## 15.3 Multiple comparisons: struck by lightning

This has been a short section, because we are skipping the technicalities and focus just on the big ideas. But this is a **VERY IMPORTANT** section to remember. When you do many tests, some of them might have low p-values just by chance and you need to adjust for this. If any gene with a low p-value is declared significant (regardless of the size of the collection of tests) you will be likely to start claiming that many purely chance results are "significant". And you do not want that.

Remember that very rare events do happen, and they are almost certain to happen if the experiment is repeated many times (by the way, this is why most of us are not afraid of dying from lighting, even if every year some people do in fact die from lightning).

When you screen 20000 genes, you are running 20000 times the experiment of the p-value and the null hypothesis. And remember the rules: for one true null hypothesis, the probability of finding a p-value $\leq 0.05$ is $0.05$. Now imagine you do that 20000 times; you are almost certain to have many p-values $\leq 0.05$. (Same thing with lightning: even if the chances of dying from lightning are $\leq \frac{1}{300000}$, with millions of people on earth, some are almost sure to die from lightning).

There are many reviews about multiple testing, FDR, etc. You might want to take a look at a three-page one by W. Noble, in *Nature Biotechnology*, 2009, 27: 1135–1136, "How does multiple testing work".

# 16 Two-way ANOVA

## 16.1 A very simple two-way ANOVA

Let us create some fake data

```
set.seed(3)
df1 <- data.frame(y = runif(8),
                  A = rep(c("a1", "a2"), 4),
                  B = rep(c("b1", "b1", "b2", "b2"), 2))


df1
```

```
##           y  A  B
## 1 0.1680415 a1 b1
## 2 0.8075164 a2 b1
## 3 0.3849424 a1 b2
## 4 0.3277343 a2 b2
## 5 0.6021007 a1 b1
## 6 0.6043941 a2 b1
## 7 0.1246334 a1 b2
## 8 0.2946009 a2 b2
```

Some summaries of data:

```
(means <- with(df1, tapply(y, list(A, B), mean)))
```

```
##           b1        b2
## a1 0.3850711 0.2547879
## a2 0.7059552 0.3111676
```

And now, several ANOVA models. We will explain each of the terms in turn in class if this is not familiar to you:

### 16.1.1 No interaction model

```
m1 <- lm(y ~ A + B, data = df1)
anova(m1)
```

```
## Analysis of Variance Table
##
## Response: y
##           Df   Sum Sq  Mean Sq F value Pr(>F)
## A          1 0.071164 0.071164  1.9312 0.2233
## B          1 0.137850 0.137850  3.7410 0.1109
## Residuals  5 0.184244 0.036849
```

The above is the anova table; we will say many more things later about that.

But for now notice also this output, that gives the estimated coefficients:

```
summary(m1)
```

```
##
## Call:
```

```
## lm(formula = y ~ A + B, data = df1)
##
## Residuals:
##       1       2       3       4       5       6       7       8
## -0.28316  0.16769  0.19628 -0.04956  0.15090 -0.03544 -0.06403 -0.08269
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.4512     0.1176   3.838   0.0121 *
## Aa2           0.1886     0.1357   1.390   0.2233
## Bb2          -0.2625     0.1357  -1.934   0.1109
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.192 on 5 degrees of freedom
## Multiple R-squared:  0.5315,Adjusted R-squared:  0.3441
## F-statistic: 2.836 on 2 and 5 DF,  p-value: 0.1502
```

We will explain it in more detail later (19), but for now remember we are using an additive model, so here we are estimating only three things from a four-means design. What three things?

### 16.1.2 Interaction model

Now all cell means are modeled:

```
m2 <- lm(y ~ A * B, data = df1)
anova(m2)
```

```
## Analysis of Variance Table
##
## Response: y
##           Df   Sum Sq  Mean Sq F value Pr(>F)
## A          1 0.071164 0.071164  1.9071 0.2394
## B          1 0.137850 0.137850  3.6942 0.1270
## A:B        1 0.034981 0.034981  0.9374 0.3878
## Residuals  4 0.149262 0.037316
```

Again, we could ask for the estimated coefficients:

```
summary(m2)
```

```
##
## Call:
## lm(formula = y ~ A * B, data = df1)
##
## Residuals:
##       1       2       3       4       5       6       7       8
## -0.21703  0.10156  0.13015  0.01657  0.21703 -0.10156 -0.13015 -0.01657
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.3851     0.1366   2.819   0.0479 *
## Aa2           0.3209     0.1932   1.661   0.1720
## Bb2          -0.1303     0.1932  -0.674   0.5370
```

```
## Aa2:Bb2      -0.2645     0.2732  -0.968   0.3878
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1932 on 4 degrees of freedom
## Multiple R-squared:  0.6204,Adjusted R-squared:  0.3358
## F-statistic:  2.18 on 3 and 4 DF,  p-value: 0.233
```

Check the interpretation of the interaction coefficient:

```
means[2, 2] -
    (means[1, 1] + coefficients(m2)[2] + coefficients(m2)[3])
```

```
##        Aa2
## -0.2645044
```

### 16.1.3 One observation per cell

What if we only had one observation per cell?

We can fit the additive model (but only 1 df left)

```
df2 <- df1[1:4, ]
m3 <- lm(y ~ A + B, data = df2)
anova(m3)
```

```
## Analysis of Variance Table
##
## Response: y
##           Df   Sum Sq  Mean Sq F value Pr(>F)
## A          1 0.084759 0.084759  0.6985 0.5568
## B          1 0.017277 0.017277  0.1424 0.7703
## Residuals  1 0.121342 0.121342
```

```
summary(m3)
```

```
##
## Call:
## lm(formula = y ~ A + B, data = df2)
##
## Residuals:
##       1       2       3       4
## -0.1742  0.1742  0.1742 -0.1742
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.3422     0.3017   1.134    0.460
## Aa2           0.2911     0.3483   0.836    0.557
## Bb2          -0.1314     0.3483  -0.377    0.770
##
## Residual standard error: 0.3483 on 1 degrees of freedom
## Multiple R-squared:  0.4568,Adjusted R-squared:  -0.6296
## F-statistic: 0.4204 on 2 and 1 DF,  p-value: 0.737
```

But we can't really fit the interaction model:

```
m4 <- lm(y ~ A * B, data = df2)
anova(m4)
```

```
## Warning in anova.lm(m4): ANOVA F-tests on an essentially perfect fit are unreliable
```

```
## Analysis of Variance Table
##
## Response: y
##           Df   Sum Sq  Mean Sq F value Pr(>F)
## A          1 0.084759 0.084759
## B          1 0.017277 0.017277
## A:B        1 0.121342 0.121342
## Residuals  0 0.000000
```

```
summary(m4)
```

```
##
## Call:
## lm(formula = y ~ A * B, data = df2)
##
## Residuals:
## ALL 4 residuals are 0: no residual degrees of freedom!
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.1680         NA      NA       NA
## Aa2           0.6395         NA      NA       NA
## Bb2           0.2169         NA      NA       NA
## Aa2:Bb2      -0.6967         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
## Multiple R-squared:      1,Adjusted R-squared:    NaN
## F-statistic:   NaN on 3 and 0 DF,  p-value: NA
```

## 16.2 Fitting a two-way ANOVA: cholesterol

The following data come from an experiment about the effects of three diets and two cholesterol-controlling drugs in the reduction of cholesterol levels (note: the response variable is change in cholesterol, so the larger the value, the larger the reduction of cholesterol). As usual, read the data and look at them. Since the author used names for the levels within each of the two factors, Diet and Drug, we do not need to transform them into factors, in contrast to what we did in section 14.1. Call the data `dcholest`.

```
dcholest <- read.table("Cholesterol.txt", header = TRUE)
```

## 16.3 Fitting a two-way ANOVA

Look at the output, which I comment below

```
## This fits the model. Pay attention to the "*"
cholestanova <- (lm(y ~ Diet*Drug, data=dcholest))
## This shows the ANOVA table. Notice the "Type II"
Anova(cholestanova)
```

```
## Anova Table (Type II tests)
##
## Response: y
##           Sum Sq Df F value    Pr(>F)
## Diet      75.453  2  29.949 3.163e-08 ***
## Drug      32.261  1  25.610 1.433e-05 ***
## Diet:Drug 48.979  2  19.441 2.348e-06 ***
## Residuals 42.830 34
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Now we are shown the 3 by 2 table of means, standard deviations, and number
## of observations
tapply(dcholest$y, list(Diet=dcholest$Diet, Drug=dcholest$Drug),
       mean, na.rm=TRUE) # means
```

```
##     Drug
## Diet         A         B
##   HF 1.7280000 -0.588400
##   M1 0.7914286  4.055714
##   M2 2.5685556  5.318250
```

```
tapply(dcholest$y, list(Diet=dcholest$Diet, Drug=dcholest$Drug),
       sd, na.rm=TRUE) # std. deviations
```

```
##     Drug
## Diet         A         B
##   HF 0.5026165 0.4956474
##   M1 0.9572549 0.7641736
##   M2 1.5181591 1.3963718
```

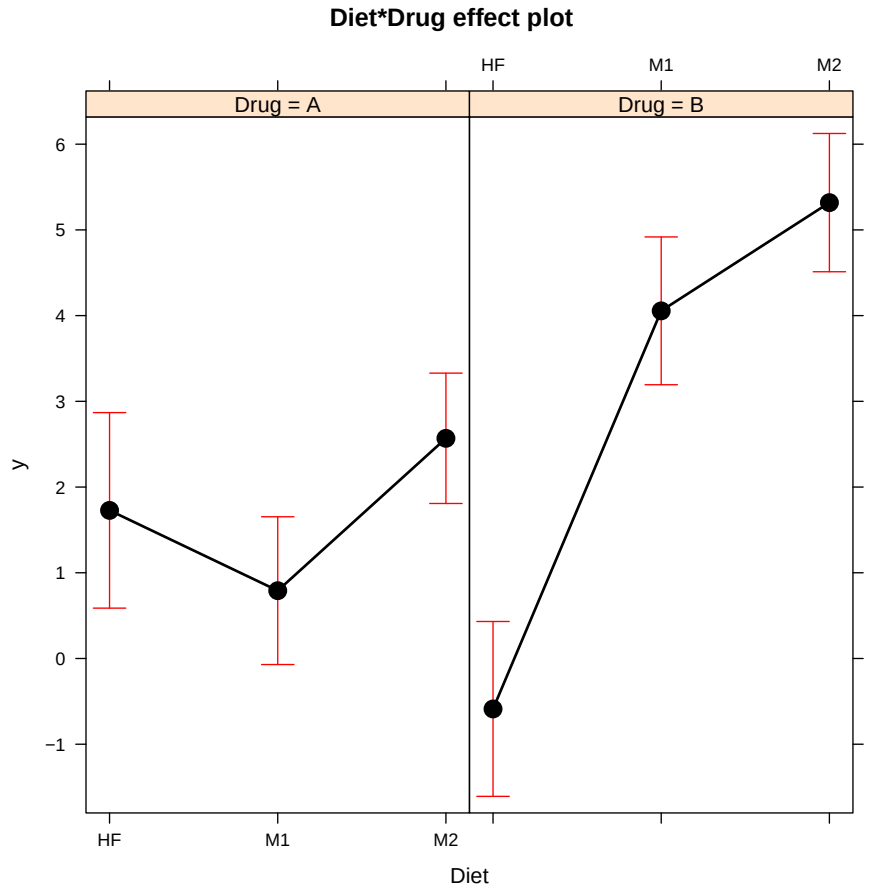
```
tapply(dcholest$y, list(Diet=dcholest$Diet, Drug=dcholest$Drug),
       function(x) sum(!is.na(x))) # counts
```

```
##     Drug
## Diet A B
##   HF 4 5
##   M1 7 7
##   M2 9 8
```

## 16.4 Interactions

We will use an "Effects plot" to look at interactions:

```
plot(allEffects(cholestanova), ask = FALSE)
```




What do you see? Do you understand what an interaction is? Do you see it in the plot? Basically, an interaction means that the effect of one variable depends on the effect of the other. In this case, even if Drug B overall leads to a larger change (decrease) in cholesterol, its effects depend on the Diet. This has practical consequences: is Drug B a better drug? It depends on the diet of the patient: for the HF (high fat) diet, Drug B is clearly worse than Drug A.

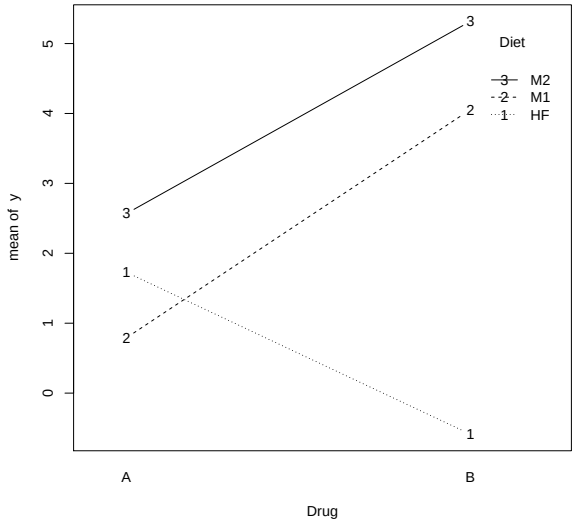
Interaction is also called "non-additivity" because the model deviates from a simple model like

$$y = Drug + Diet$$

as the effect of Drug depends on the value of Diet (or the other way around). The phenomenon of interaction should be familiar to you: it is very common in life in general, and in biology you might have previously seen at as epistasis in genetics.

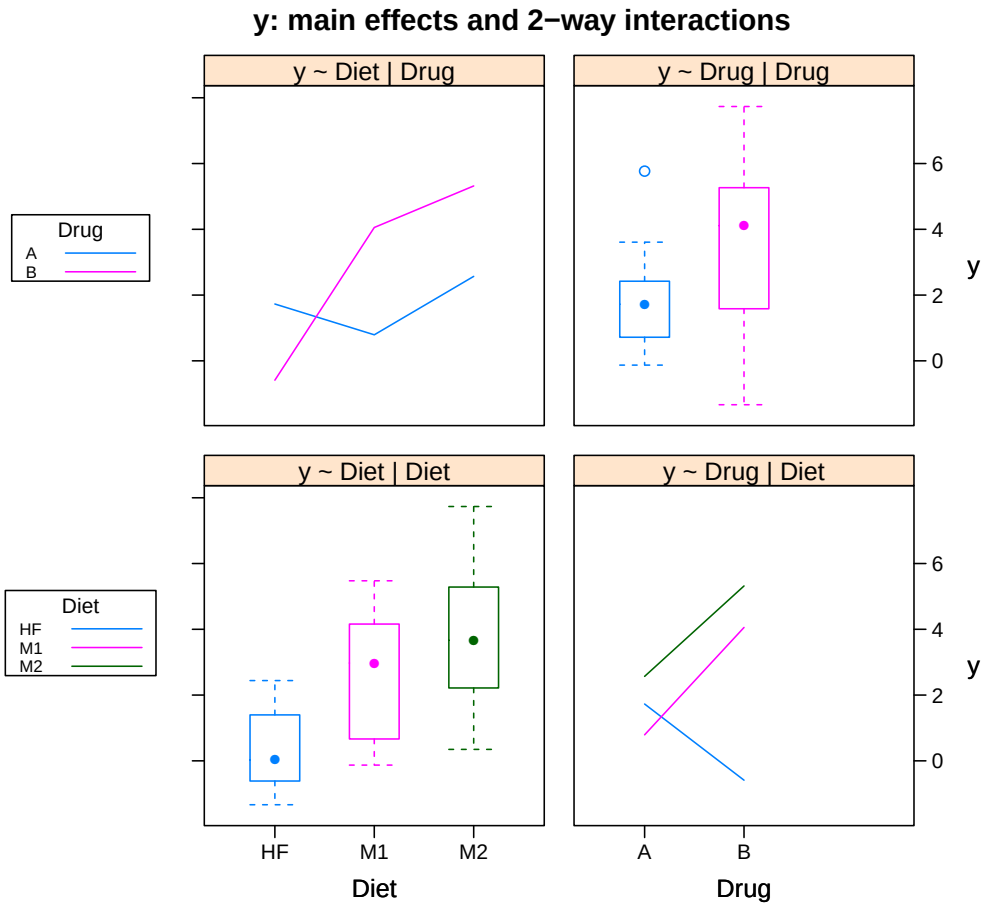
You can also see interaction plots using other functions from R. For example:

```
with(dcholest, interaction.plot(Drug, Diet, y, type = "b"))
```




or using the *HH* package

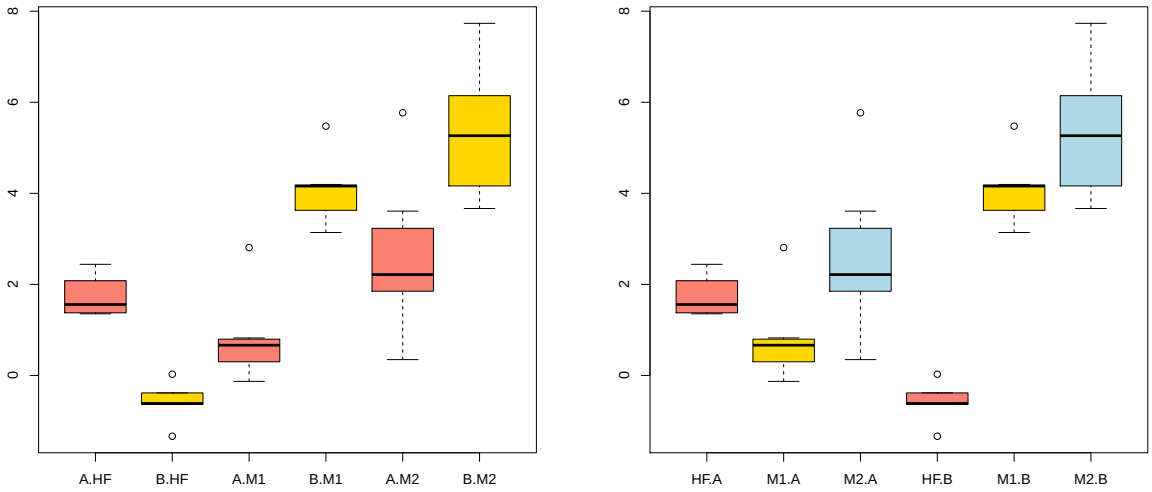
```
library(HH)
## Loading required package: lattice
## Loading required package: grid
## Loading required package: latticeExtra
## Loading required package: RColorBrewer
## Loading required package: gridExtra
##
## Attaching package: 'HH'
## The following objects are masked from 'package:car':
##
##     logit, vif
interaction2wt(y ~ Diet + Drug, data = dcholest)
```




Notice how this last figure displays both main effects and interactions. So even if the main effect of Drug B is to lead to a larger change in cholesterol (as you can see in the upper right panel), Drug B actually leads to much smaller change in cholesterol if given to patients with diet HF (as seen in the bottom right panel). In fact, under Drug A, it seems that diet HF is actually slightly better than diet M1 (as seen in the bottom right panel or in the effects plot).

Finally, a boxplot can also help show the interaction. I will use two different ones, that differ by the order in which factors are specified (one or the other might be easier to decode visually):

```
boxplot(y ~ Drug * Diet, data = dcholest, col = c("salmon", "gold"))
boxplot(y ~ Diet * Drug, data = dcholest,
        col = c("salmon", "gold", "lightblue"))
```




Given these results (the strong interaction, that can even revert effects of one factor), it makes little sense to report any global main effects and we would rarely be interested in interpreting the significance (or not) of the Diet or Drug term. In general, **in the presence of interactions, we often refrain from interpreting main effects**[13].

## 16.5 An ANOVA without interactions

Could we fit a model without interactions? Yes, of course. The idea is to change the "*" by a "+" (and we will see that again when we deal with multiple regression et al. in section 18.1).

```
amodelnoint <- (lm(y ~ Diet + Drug, data=dcholest))
Anova(amodelnoint)

## Anova Table (Type II tests)
##
## Response: y
##           Sum Sq Df F value    Pr(>F)
## Diet      75.453  2  14.793 2.046e-05 ***
## Drug      32.261  1  12.650  0.001074 **
## Residuals 91.809 36
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

There are, however, good reasons to start fitting a model **with** interactions first, and **only** if there are no interactions, fit a simpler, additive model.

## 16.6 The order of factors

Let's pretend there are no interactions. We can do that by creating a data set without the "HF" subjects.

[13] Properly formulated, which also generally involves using other types of contrasts —such as contr.sum, in R parlance— marginal tests in the presence of interactions, what are called Type III, can make sense, but are not always of interest. See also footnote 14 in section 16.6 for some entries and references.

```
dcholest2 <- subset(dcholest, subset = Diet != "HF")
```

Now do a two-way ANOVA:

```
cholest2anova <- (lm(y ~ Diet*Drug, data = dcholest2))
Anova(cholest2anova)
```

```
## Anova Table (Type II tests)
##
## Response: y
##           Sum Sq Df F value    Pr(>F)
## Diet      17.879  1 11.7483  0.001967 **
## Drug      68.809  1 45.2150 3.216e-07 ***
## Diet:Drug  0.507  1  0.3335  0.568417
## Residuals 41.089 27
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

So no evidence whatsoever of interactions. For simplicity, we can go and refit the model without the interaction.We will actually fit two models, which differ only by the order in which we give Diet and Drug in the formula and we will call them lm1 and lm2

```
lm1 <- lm(y ~ Diet + Drug, data = dcholest2)
lm2 <- lm(y ~ Drug + Diet, data = dcholest2)
```

Look at the ANOVA tables:

```
anova(lm1)
```

```
## Analysis of Variance Table
##
## Response: y
##           Df Sum Sq Mean Sq F value    Pr(>F)
## Diet       1 15.897  15.897  10.701  0.002842 **
## Drug       1 68.809  68.809  46.318 2.156e-07 ***
## Residuals 28 41.597   1.486
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(lm2)
```

```
## Analysis of Variance Table
##
## Response: y
##           Df Sum Sq Mean Sq F value    Pr(>F)
## Drug       1 66.827  66.827  44.983 2.793e-07 ***
## Diet       1 17.879  17.879  12.035  0.001708 **
## Residuals 28 41.597   1.486
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As you can see, the F statistic and the p-value are different!!! What gives here?

Now, use Type II sums of squares:

```
Anova(lm1)
```

```
## Anova Table (Type II tests)
```

```
##
## Response: y
##           Sum Sq Df F value    Pr(>F)
## Diet      17.879  1  12.035  0.001708 **
## Drug      68.809  1  46.318 2.156e-07 ***
## Residuals 41.597 28
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Anova(lm2)
```

```
## Anova Table (Type II tests)
##
## Response: y
##           Sum Sq Df F value    Pr(>F)
## Drug      68.809  1  46.318 2.156e-07 ***
## Diet      17.879  1  12.035  0.001708 **
## Residuals 41.597 28
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Nothing changes between those two. But if you look carefully, the F value (and p-value) of the Type II Sums of Squares ANOVA table are the same as those for the term that enters last in the Type I (those produced via `anova`, without a capital "A").

This sounds crazy, irrelevant, and a huge wast of time. But it ain't. This is an **extremely common phenomenon** when the design is not perfectly balanced (with categorical independent variables) or there are correlations (with continuous covariates, as in regression). What is happening?

- Type II sums of squares (similar to t-statistics from a linear model) show what that term contributes, **given all the rest** are already in the model. In other words, given all the other terms (that do not include this term) have already been taken into account. This is actually the output we would get from comparing two models, one with all terms, and one with all terms except the term in question. (Always assuming interactions with the term in question are zero). The package *car*, by default, gives you this via `Anova`. I routinely use `Anova`.
- Type I (or sequential) sums of squares do not. They are sequential, in the order shown in the output. R, by default, gives you this via `anova`.

Biologically, that order can make a difference makes a lot of sense. Think about it. And this, of course, affects models with two, three, ... factors. Always pay attention to what it is you are being reported (and beware that the defaults used by R need not the same as those used by SPSS, SAS, etc.)[14].

Some of this might still seem mysterious. Think about doing a regression of body height on the length of both the left and right arms. And think about how unbalanced data, with categorical independent variables, is somewhat similar to inducing a correlation between variables. We will discuss this in class.

[14] The type of sums of squares used (and, actually, the type of hypothesis tested) is a debated topic. The book by John Fox "Applied regression analysis and generalized linear models" contains a great discussion (his book with S. Weisberg "An R companion to applied regression", one of the recommended books, does too). An email-length discussion of these topics can be found here https://stat.ethz.ch/pipermail/r-help/2006-August/111854.html and https://stat.ethz.ch/pipermail/r-help/2006-August/111927.html.

## 16.7 Does order always matter?

Nope. When the design is balanced, order does not matter[15].

This is slightly more advanced material. Skip it on first reading. If you want a very simple message: **unless you know what your are doing and/or you know your data fulfills certain properties, always expect order to matter**.

If you want to continue reading, let's proceed with some details then. The following is an example. For the sake of the exposition, I will here simulate the data, so everything is clear and in the open.

```
set.seed(1)
sex <- factor(rep(c("Male", "Female"), c(20, 20)))
drug <- factor(rep(rep(c("A", "B"), c(10, 10)), 2))
y <- rep(c(10, 13, 12, 16), rep(10, 4))
y <- y + rnorm(length(y), sd = 1.5)
y.data <- data.frame(y, sex, drug)
```

First, some basic stats about those data. Notice the perfect balance:

```
with(y.data, tapply(y, list(sex, drug), function(x) sum(!is.na(x))))
```

```
##         A  B
## Female 10 10
## Male   10 10
```

```
with(y.data, tapply(y, list(sex, drug), mean))
```

```
##               A        B
## Female 11.79949 16.18110
## Male   10.19830 13.37327
```

Just by eye, it seems the difference between sexes is around 2, and the difference between drugs of about 4. And no, there is no interaction:

```
summary(lm(y ~ sex * drug, data = y.data))
```

```
##
## Call:
## lm(formula = y ~ sex * drug, data = y.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6953 -0.6854  0.1639  0.9228  2.1946
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    11.7995     0.4322  27.304  < 2e-16 ***
## sexMale        -1.6012     0.6112  -2.620   0.0128 *
## drugB           4.3816     0.6112   7.169 1.97e-08 ***
## sexMale:drugB  -1.2066     0.8643  -1.396   0.1712
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.367 on 36 degrees of freedom
```

[15] Technically, orthogonality of the design matrix does not require identical cell counts; if row/column counts are proportional, for instance, we should be OK. But for simplicity, we will use a nicely balanced example here.

```
## Multiple R-squared:  0.7436,Adjusted R-squared:  0.7222
## F-statistic:  34.8 on 3 and 36 DF,  p-value: 9.746e-11
```

Fit two models, simply changing the order (we assume no interaction, as shown above).

```
m1 <- lm(y ~ sex + drug, data = y.data)
m2 <- lm(y ~ drug + sex, data = y.data)
```

And we also fit two small models, one only with sex, the other only with drug:

```
msex <- lm(y ~ sex, data = y.data)
mdrug <- lm(y ~ drug, data = y.data)
```

Now, the output for the coefficients for m1 and m2 is the same (these are always the coefficients as if entered last in the model):

```
summary(m1)
```

```
##
## Call:
## lm(formula = y ~ sex + drug, data = y.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.9970 -0.7100  0.0357  0.8676  2.4963
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  12.1012     0.3790  31.927  < 2e-16 ***
## sexMale      -2.2045     0.4377  -5.037 1.26e-05 ***
## drugB         3.7783     0.4377   8.633 2.15e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.384 on 37 degrees of freedom
## Multiple R-squared:  0.7297,Adjusted R-squared:  0.7151
## F-statistic: 49.95 on 2 and 37 DF,  p-value: 3.079e-11
```

```
summary(m2)
```

```
##
## Call:
## lm(formula = y ~ drug + sex, data = y.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.9970 -0.7100  0.0357  0.8676  2.4963
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  12.1012     0.3790  31.927  < 2e-16 ***
## drugB         3.7783     0.4377   8.633 2.15e-10 ***
## sexMale      -2.2045     0.4377  -5.037 1.26e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 1.384 on 37 degrees of freedom
## Multiple R-squared:  0.7297,Adjusted R-squared:  0.7151
## F-statistic: 49.95 on 2 and 37 DF,  p-value: 3.079e-11
```

So nothing new up to here. Now look at what happens if we get the coefficients for the small models, those with only sex or only drug:

```
summary(msex)
```

```
##
## Call:
## lm(formula = y ~ sex, data = y.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.9743 -1.9275 -0.3339  1.9228  4.0477
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  13.9903     0.5302   26.39  < 2e-16 ***
## sexMale      -2.2045     0.7498   -2.94  0.00556 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.371 on 38 degrees of freedom
## Multiple R-squared:  0.1853,Adjusted R-squared:  0.1639
## F-statistic: 8.645 on 1 and 38 DF,  p-value: 0.005556
```

```
summary(mdrug)
```

```
##
## Call:
## lm(formula = y ~ drug, data = y.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.0992 -1.1956  0.0094  1.1359  3.2608
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  10.9989     0.3965  27.741  < 2e-16 ***
## drugB         3.7783     0.5607   6.738 5.57e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.773 on 38 degrees of freedom
## Multiple R-squared:  0.5444,Adjusted R-squared:  0.5324
## F-statistic: 45.41 on 1 and 38 DF,  p-value: 5.569e-08
```

In both cases, the estimate is the same from the model with the two factors, or with only a single factor. For example, the differences between sexes are of about 2.2 (the coefficient that says "sexMale") and the differences between drugs of about 3.8 (the coefficient that says "drugB"). However, the standard error and, thus, the t value and the p-value change.

Again, the key is to understand that even if the coefficient does not change whether or not the other factor is included in the model (and it does not change because there is complete balance here), the t statistic and the p-value do change. Why? Because the other factor explains a large part of variance, and thus makes the residual standard error much smaller if we include it in the model.

And what about the ANOVA tables?

```
anova(m1)
```

```
## Analysis of Variance Table
##
## Response: y
##           Df  Sum Sq Mean Sq F value    Pr(>F)
## sex        1  48.599  48.599  25.371 1.258e-05 ***
## drug       1 142.754 142.754  74.527 2.149e-10 ***
## Residuals 37  70.873   1.915
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(m2)
```

```
## Analysis of Variance Table
##
## Response: y
##           Df  Sum Sq Mean Sq F value    Pr(>F)
## drug       1 142.754 142.754  74.527 2.149e-10 ***
## sex        1  48.599  48.599  25.371 1.258e-05 ***
## Residuals 37  70.873   1.915
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Order (when we include both factors, of course) does not change anything. Why? Because the contributions of each factor do not depend at all on the other (i.e., the Mean Squares of each factor does not depend on the other). And since the F is the ratio of the Mean Squares of the factor over the Mean Squares of the residuals (and this is whatever is left after we have fitted everything), the order does not affect the F statistic or the p-value.

Of course, an "Anova" (Type II tests) would show the same:

```
Anova(m1)
```

```
## Anova Table (Type II tests)
##
## Response: y
##            Sum Sq Df F value    Pr(>F)
## sex        48.599  1  25.371 1.258e-05 ***
## drug      142.754  1  74.527 2.149e-10 ***
## Residuals  70.873 37
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

To understand this better, look at the anova tables for the models with only one factor:

```
anova(msex)
```

```
## Analysis of Variance Table
##
```

```
## Response: y
##           Df  Sum Sq Mean Sq F value   Pr(>F)
## sex        1  48.599  48.599  8.6447 0.005556 **
## Residuals 38 213.627   5.622
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(mdrug)
```

```
## Analysis of Variance Table
##
## Response: y
##           Df Sum Sq Mean Sq F value    Pr(>F)
## drug       1 142.75 142.754  45.406 5.569e-08 ***
## Residuals 38 119.47   3.144
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Notice how the Mean Sq for each factor is the same as in the previous tables. So the Mean Squares for Sex do not depend on whether or not drug is in the model. But the F statistic (and the p-value) do change a lot. Why? Because what changes a lot are the Mean Sq. of the residuals. And why is that? Because the other factor, the one we have not included, does indeed explain a lot of variability, but in these two last tables, since the other factor is not in the model, that variability is included now in the error term.

So, to summarize: when there is balance, order does not change a thing if we include both factors in the model. However, having or not the other factor in the model can make a difference for the standard errors, the residual standard errors, and thus the p-values.

## 16.8 One observation per cell

So far, we have had more than one observation for each combination of levels (i.e., more than one observation per cell, where cell means each of the "places" or "boxes" in a table that shows the combinations of treatments). What if we only had one?

Let us simulate some data:

```
set.seed(3)
df1 <- data.frame(y = runif(6),
                  A = rep(c("a1", "a2", "a3"), 2),
                  B = rep(c("b1", "b2"), rep(3, 2)))
df1
```

```
##           y  A  B
## 1 0.1680415 a1 b1
## 2 0.8075164 a2 b1
## 3 0.3849424 a3 b1
## 4 0.3277343 a1 b2
## 5 0.6021007 a2 b2
## 6 0.6043941 a3 b2
```

Some summaries of data:

```
(means <- with(df1, tapply(y, list(A, B), mean)))
```

```
##           b1        b2
```

```
## a1 0.1680415 0.3277343
## a2 0.8075164 0.6021007
## a3 0.3849424 0.6043941
```

We can fit the additive model (but only 1 df left)

```
m1 <- lm(y ~ A + B, data = df1)
anova(m1)
```

```
## Analysis of Variance Table
##
## Response: y
##           Df   Sum Sq  Mean Sq F value Pr(>F)
## A          2 0.209224 0.104612  3.9552 0.2018
## B          1 0.005030 0.005030  0.1902 0.7053
## Residuals  2 0.052898 0.026449
```

```
summary(m1)
```

```
##
## Call:
## lm(formula = y ~ A + B, data = df1)
##
## Residuals:
##        1        2        3        4        5        6
## -0.05089  0.13166 -0.08077  0.05089 -0.13166  0.08077
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.21893    0.13279   1.649    0.241
## Aa2          0.45692    0.16263   2.810    0.107
## Aa3          0.24678    0.16263   1.517    0.268
## Bb2          0.05791    0.13279   0.436    0.705
##
## Residual standard error: 0.1626 on 2 degrees of freedom
## Multiple R-squared:  0.802,Adjusted R-squared:  0.505
## F-statistic:   2.7 on 3 and 2 DF,  p-value: 0.2818
```

But we can't really fit the interaction model:

```
m2 <- lm(y ~ A * B, data = df1)
anova(m2)
```

```
## Warning in anova.lm(m2):  ANOVA F-tests on an essentially perfect fit are unreliable
```

```
## Analysis of Variance Table
##
## Response: y
##           Df   Sum Sq  Mean Sq F value Pr(>F)
## A          2 0.209224 0.104612
## B          1 0.005030 0.005030
## A:B        2 0.052898 0.026449
## Residuals  0 0.000000
```

```
summary(m2)
```

```
##
```

```
## Call:
## lm(formula = y ~ A * B, data = df1)
##
## Residuals:
## ALL 6 residuals are 0: no residual degrees of freedom!
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.16804         NA      NA       NA
## Aa2          0.63947         NA      NA       NA
## Aa3          0.21690         NA      NA       NA
## Bb2          0.15969         NA      NA       NA
## Aa2:Bb2     -0.36511         NA      NA       NA
## Aa3:Bb2      0.05976         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
## Multiple R-squared:      1,Adjusted R-squared:    NaN
## F-statistic:   NaN on 5 and 0 DF,  p-value: NA
```

Do you understand what is going on?

## 16.9 ANOVA/linear models with more than two factors

We will present no examples, but life is filled with them. Think about cholesterol: to the experiment with drug and diet add a third factor: an exercise program. Or maybe a fourth factor too: a stress reduction program. Or maybe ....

## 16.10 Multiple comparisons of means in two-way ANOVA

Can they be done? Yes. You can use the `glht` or the TukeyHDS functions directly. Or you fit the model using `aov` (not `lm`), and use the package *HH*, and ask for the "MMC plot". We will not pursue this any further here (among other questions you should ask yourself, at what levels of one variable will you be comparing the other? How does lack of balance affect the contrasts? Do you really want all possible contrasts?) A good place to start reading on these issues is chapter 14 (and section 5.3.2) of Everitt and Hothorn's "A handbook of statistical analysis using R, 2nd ed".

## 16.11 Nonparametric alternatives

Are there nonparametric versions of the above procedures? For the one-way ANOVA the Kruskal-Wallis test is popular. For two-way designs with one observation per cell the Friedman test and the Quade test. But testing interactions is not easy; one needs to use more sophisticated approaches as in permutation tests conditioning on permuting only within rows or columns, etc. We will not pursue this any further.

# 17 Simple linear regression

This is another form of a linear model. But now, the independent variable is continuous. So we will fit a line:

$$Y = \alpha + \beta X + \epsilon$$

where $Y$ is, as usual, the dependent variable, $X$ the independent, $\beta$ is the slope and $\alpha$ the intercept (this is just the equation for a line). The simple linear regression procedure will estimate $\alpha$ and $\beta$, finding values $(\hat{\alpha}, \hat{\beta})$ that produce a **best fitting line** (note: it is a line, not an arbitrary curve).

We will use a subset of data from the AnAge data set (Animal Ageing and Longevity Database) (accessed on 2014-08-19) from http://genomics.senescence.info/species/. This file contains longevity, metabolic rate, body mass, and a variety of other life history variables. The data I provide you are a small subset that includes only some birds and reptiles.

Read the full data and call it `anage_a_r` (the a and r stand for aves and reptilia, the proper Class names).

We want to take the log of all the relevant continuous variables (yes, you would not know this before hand, but I do, so create those new variables now to avoid going back later)[16].

```
anage_a_r$logMetabolicRate <- log(anage_a_r$Metabolic.rate..W.)
anage_a_r$logBodyMass <- log(anage_a_r$Body.mass..g.)
anage_a_r$logLongevity <- log(anage_a_r$Maximum.longevity..yrs.)
```

For now, we will only use the birds. So use subsetting to keep only birds and call it `anage_a`

We want to model metabolic rate as a function of body mass (note that this data set is rather nice, because column names are nicely labeled and include information about units). **Beware:** biologically, what we are going to do is not really correct, as the data are not independent (species share common ancestors, and they are related in varying degrees, as any phylogenetic tree would show you, and as you should be able to tell from looking at the names of some species). What we are doing here is just for the sake of the example, and because this is a nice set of data[17].

```
metab <- lm(Metabolic.rate..W. ~ Body.mass..g., data = anage_a)
summary(metab)
```

```
##
## Call:
## lm(formula = Metabolic.rate..W. ~ Body.mass..g., data = anage_a)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.2390 -0.3386 -0.2095  0.1578  3.8380
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.5300123  0.0694005   7.637 1.74e-12 ***
## Body.mass..g. 0.0025673  0.0001133  22.663  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

[16] Creating these new variables is not really necessary in general for fitting models. But some functions from the *HH* package lead to problems if we don't.

[17] This can be done correctly, incorporating phylogenetic information in the regression model, but this is way out of the scope of this class. It is a really fascinating topic, though!

```
##
## Residual standard error: 0.7975 on 164 degrees of freedom
##   (1020 observations deleted due to missingness)
## Multiple R-squared:  0.758,Adjusted R-squared:  0.7565
## F-statistic: 513.6 on 1 and 164 DF,  p-value: < 2.2e-16
```

The row of the output that says "(Intercept)" gives you the estimate of the intercept. The t-statistic (under "t value") is testing that the intercept is zero. And it is not. But tests about the intercept are rarely interesting (except for cases with a natural and meaningful 0). The second line is more interesting: that is the slope, how much metabolic rate increases per unit increase in body mass (of course, to interpret this we need to know the units!). And the t-statistic tests if the slope is 0. There is certainly strong evidence that Metabolic rate increases with body mass.

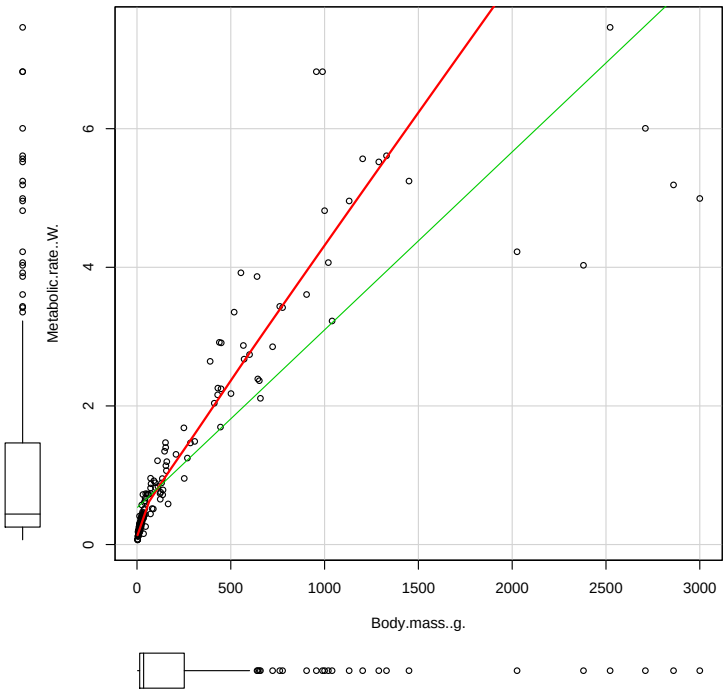
Do you know what "R-squared" refers to? And the rest of the output?

By the way, did you see the note about missingness? Do you know what that means?

## 17.1 And how does it look like

Eh!!! We should probably have plotted the data as the first thing. A couple of plots will be good here. First, let's do a scatterplot (you might want to not show the spread [18]):

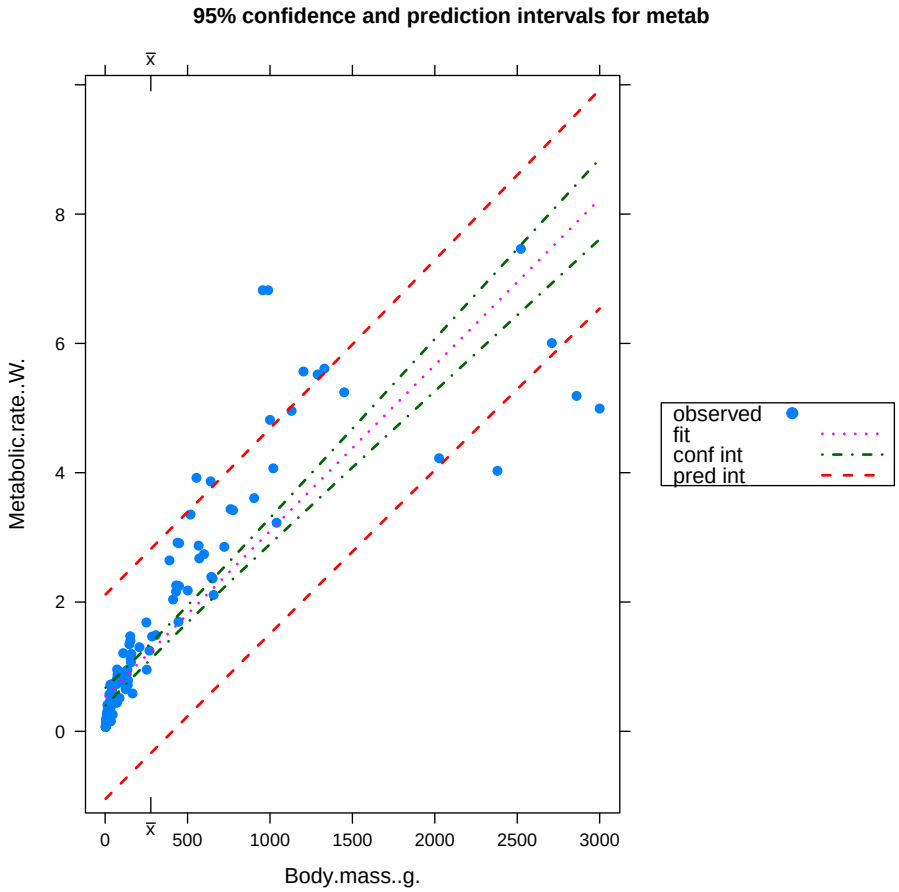
```
scatterplot(Metabolic.rate..W. ~ Body.mass..g.,
            smooth = TRUE,
            spread = FALSE,  data = anage_a)
```




The second plot we will do requires you to load *HH*

```
library(HH)
ci.plot(metab)
```

[18] I find the spread information to be confusing. But I like to leave the smoothed line, as it can help me see errors in the model specification.




(and this shows confidence and prediction intervals for the linear model).

### 17.1.1 Transforming the data

Hummm.... Those plots do not look good. OK, let's refit a model, but this time let's transform both the dependent and independent variables with a log (why a log? theory and previous empirical evidence from the field of allometry and life history suggest that it is a reasonable way to go).

First fit the model: [19]

```
metablog <- lm(logMetabolicRate ~ logBodyMass, data = anage_a)
summary(metablog)
```

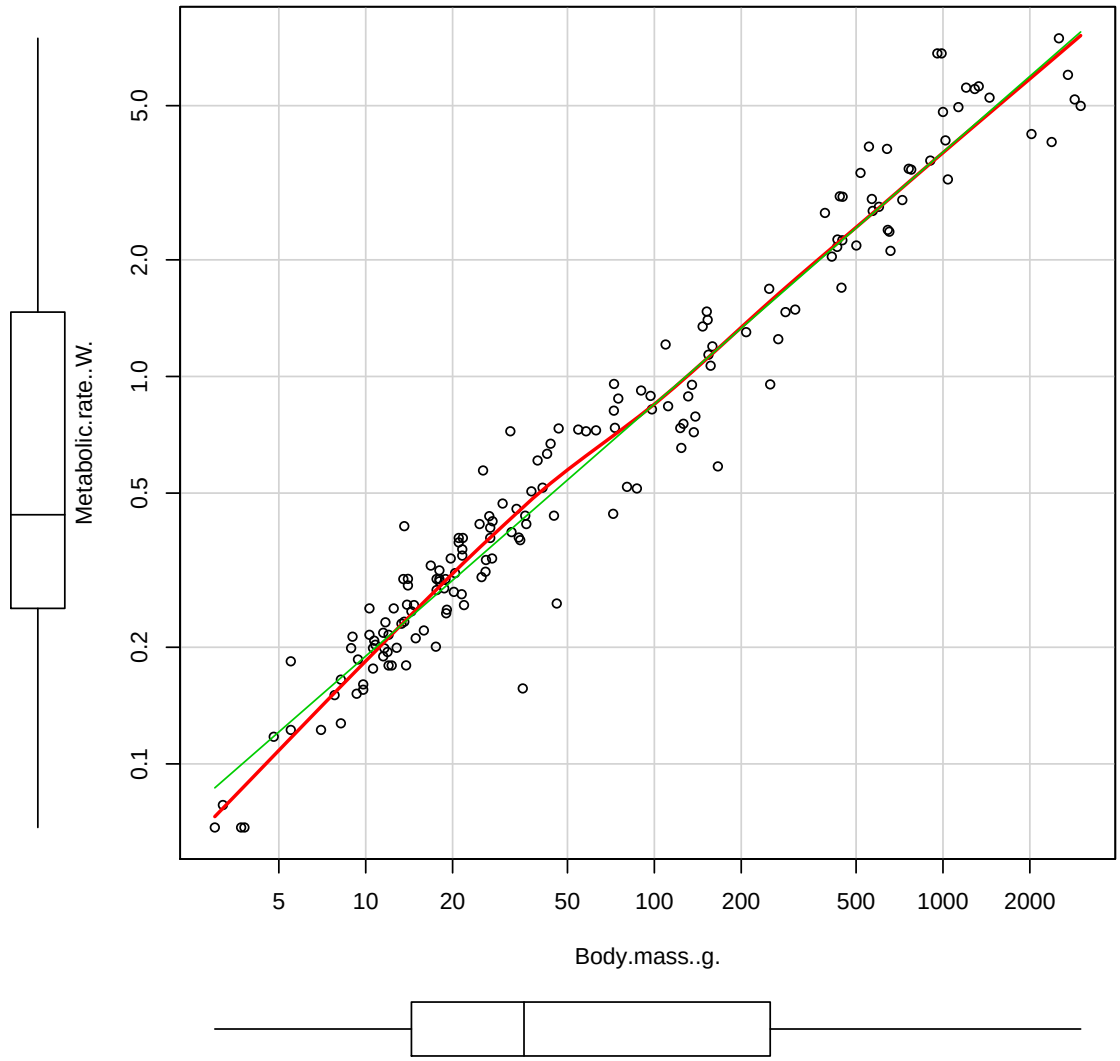
```
##
## Call:
## lm(formula = logMetabolicRate ~ logBodyMass, data = anage_a)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.00686 -0.14349  0.01545  0.16584  0.61638
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3.15949    0.04895  -64.55   <2e-16 ***
```

[19] You could have fitted the model as
`metablog <- lm(log(Metabolic.rate..W.) ~ log(Body.mass..g.), data = anage_a)`
and that would have been fine. But then, functions `ci.plot` and `ancova` from *HH* choke.

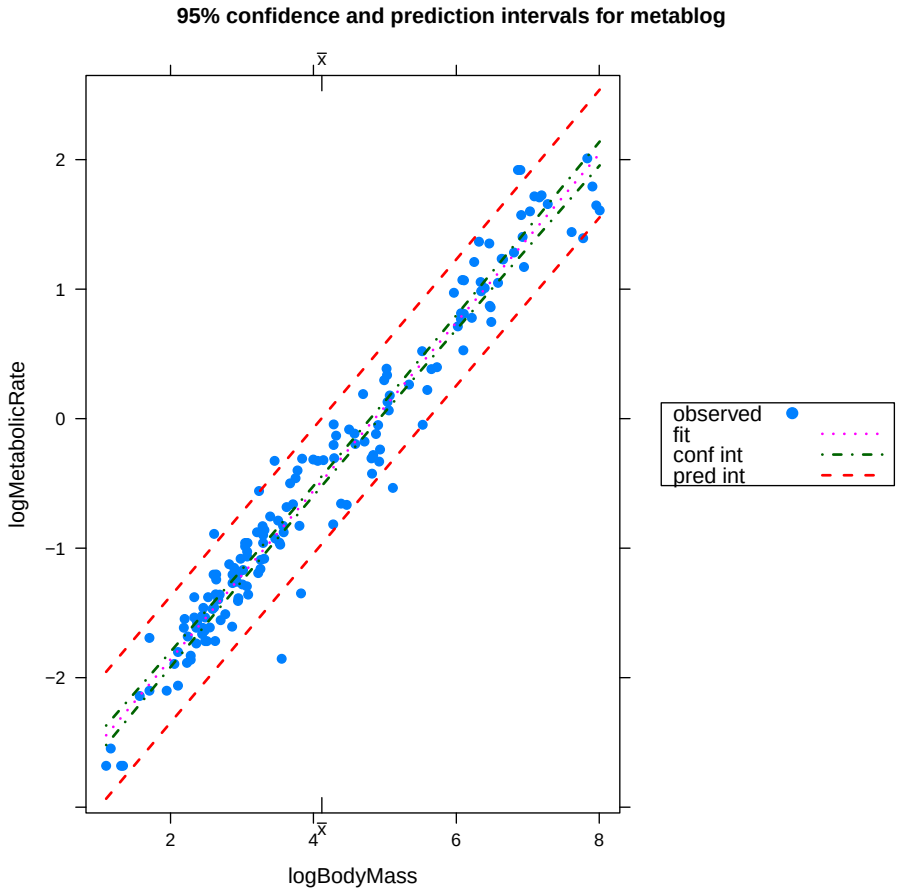
```
## logBodyMass  0.65037    0.01095   59.38   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2452 on 164 degrees of freedom
##   (1020 observations deleted due to missingness)
## Multiple R-squared:  0.9556,Adjusted R-squared:  0.9553
## F-statistic:  3527 on 1 and 164 DF,  p-value: < 2.2e-16
```

Now the two plots:

```
scatterplot(Metabolic.rate..W. ~ Body.mass..g., log = "xy",
            smooth = TRUE, spread = FALSE,
            data = anage_a)
```




```
ci.plot(metablog)
```




These are both much, much better. We will address this issue more formally below (section 23). Notice that the call to `scatterplot` uses the original variables but the axis are in log-scale, which is nicer than directly plotting (in linear scale) the log-transformed variables: you can see the original values.

But this was a particularly simple example since I told you how to transform the data. You should be asking yourself: how do I know what transformation to use? Often, theory (should allometric patterns scale with the log? shouldn't we use the square root for phenomena that take place on surfaces? etc) can guide us. Otherwise, there are procedures to try to identify transformations, including some diagnostic plots that can help (e.g., component+residual plots, section 23).

# 18 Multiple regression

In the previous section we had

$$Y = \alpha + \beta X + \epsilon$$

Now we can have two or more independent variables:

$$Y = \alpha + \beta_1 X_1 + \beta_2 X_2 + \ldots + \epsilon$$

I will use a dataset that is a small subset from the original `cystfibr` data set from package *ISwR* (by Peter Dalgaard; this package is also material to accompany Dalgaard's book "Introductory statistics with R")

Import the dataset.

```
cystfibr2 <- read.table("CystFibr2.txt", header = TRUE)
```

The meaning of the variables is (this is copied verbatim from the help of the original dataset):

```
'age' a numeric vector, age in years.
'sex' a numeric vector code, 0: male, 1:female.
'height' a numeric vector, height (cm).
'weight' a numeric vector, weight (kg).
'pemax' a numeric vector, maximum expiratory pressure.
```

For the multiple regression we model

$$pemax = \alpha + \beta_1 age + \beta_2 height + \beta_3 weight + \epsilon$$

```
mcyst <- lm(pemax ~ age + height + weight, data=cystfibr2)
summary(mcyst)
```

```
##
## Call:
## lm(formula = pemax ~ age + height + weight, data = cystfibr2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -43.675 -21.566   3.229  16.274  48.068
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 64.65555   82.40935   0.785    0.441
## age          1.56755    3.14363   0.499    0.623
## height      -0.07608    0.80278  -0.095    0.925
## weight       0.86949    0.85922   1.012    0.323
##
## Residual standard error: 27.41 on 21 degrees of freedom
## Multiple R-squared:  0.4118,Adjusted R-squared:  0.3278
## F-statistic: 4.901 on 3 and 21 DF,  p-value: 0.009776
```

You should be able to interpret all output without problems.

Now, use ANOVA tables with Type I sums of squares and Type II sums of squares:

```
anova(mcyst)
```

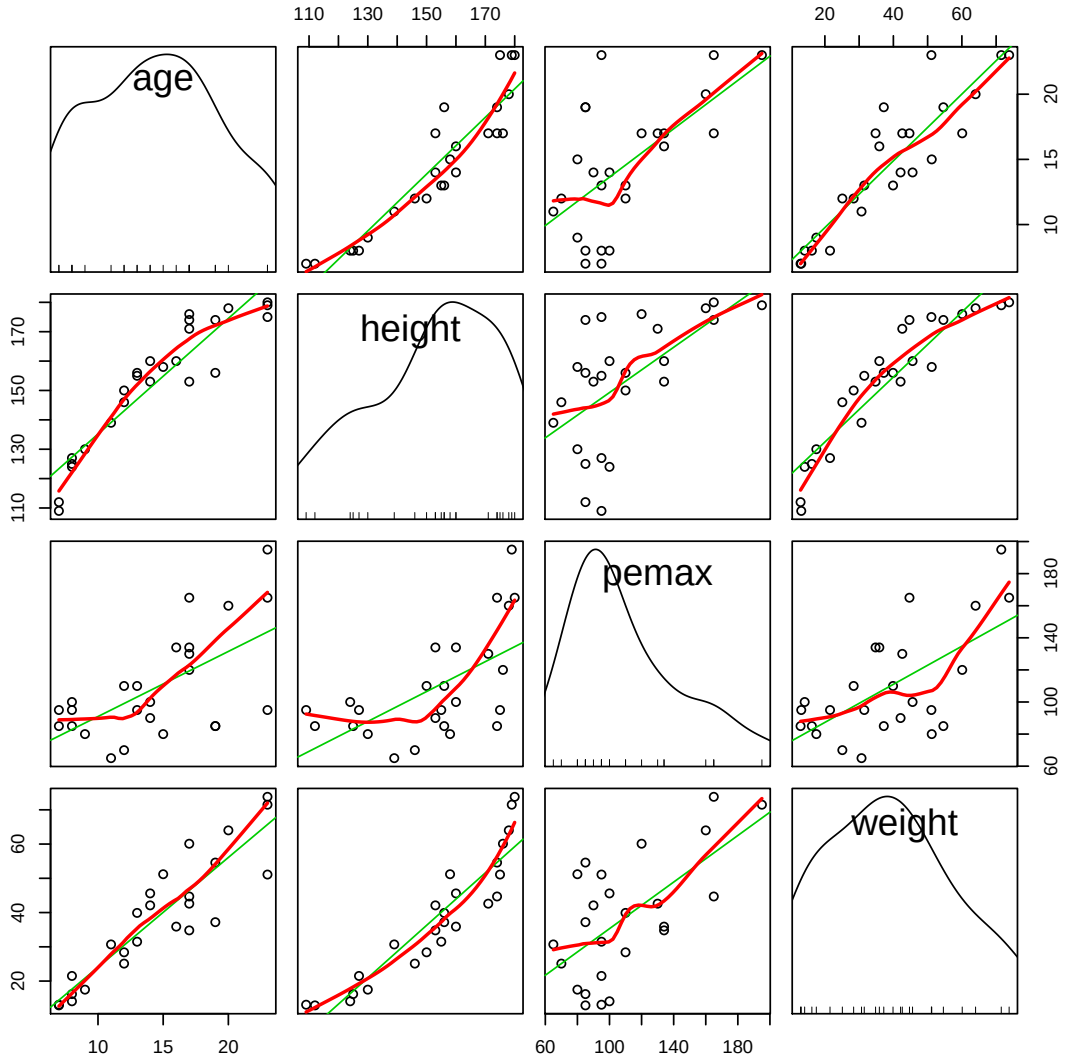
```
## Analysis of Variance Table
##
## Response: pemax
##           Df  Sum Sq Mean Sq F value   Pr(>F)
## age        1 10098.5 10098.5 13.4371 0.001441 **
## height     1   182.3   182.3  0.2426 0.627427
## weight     1   769.6   769.6  1.0240 0.323082
## Residuals 21 15782.2   751.5
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Anova(mcyst)
```

```
## Anova Table (Type II tests)
##
## Response: pemax
##            Sum Sq Df F value Pr(>F)
## age         186.9  1  0.2486 0.6232
## height        6.8  1  0.0090 0.9254
## weight      769.6  1  1.0240 0.3231
## Residuals 15782.2 21
```

Are you surprised? Age seems highly significant with the sequential sums of squares (when entered first and using `anova`, so Type I) but not when we test it after all other terms in the model (`Anova`, or Type II). Why? One possible explanation is that there is correlation between explanatory variables, and that the information of age relevant for predicting pmax is already contained in the height and weight. That age, height, and weigth are correlated is easy to check with a scatterplot matrix:

```
scatterplotMatrix( ~ age+height+pemax+weight, smooth = TRUE,
                   spread = FALSE,  data = cystfibr2)
```




In fact, if you refit the model and now put height first, and do a sequential test you find ... that height seems significant:

```
anova(lm(pemax ~ height + weight + age, data = cystfibr2))

## Analysis of Variance Table
##
## Response: pemax
##           Df  Sum Sq Mean Sq F value   Pr(>F)
## height     1  9634.6  9634.6 12.8200 0.001763 **
## weight     1  1228.9  1228.9  1.6352 0.214935
## age        1   186.9   186.9  0.2486 0.623214
## Residuals 21 15782.2   751.5
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

And similar if you place weight first.

```
anova(lm(pemax ~ weight + height + age, data = cystfibr2))

## Analysis of Variance Table
```

```
##
## Response: pemax
##           Df  Sum Sq Mean Sq F value   Pr(>F)
## weight     1 10827.2 10827.2 14.4067 0.001058 **
## height     1    36.4    36.4  0.0484 0.827949
## age        1   186.9   186.9  0.2486 0.623214
## Residuals 21 15782.2   751.5
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Is this a problem? Well, you are trying to model pemax as a function of three variables, but those three variables are very highly correlated among themselves. Is this a common phenomenon: yes, it is rather common.

## 18.1 Interactions between continuous variables

Can we add interactions? Yes, of course. Interactions between continuous variables, however, are harder to visualize (they represent curved surfaces, because the slope of one of the variable changes as the other variable changes, whereas an additive model is just a plane —or hyperplane)[20].

[20] If you want to play around with a regression plane or regression surfaces, go to "Graphs", "3D graph", "3D scatterplot". In options, you can choose a plane or three different surfaces —you might need to play with the degrees of freedom if you get errors. You can zoom in and out with the mouse wheel and move/rotate the figure. Try doing that during your TFM defense! Of course, that allows only for up to one dependent variable and two independent ones: our brains do not seem ready for 4D and higher.

# 19 Anova tables from `lm` et al.: understanding the coefficients and parameters

We've said this: ANOVAs are a type of linear models. Thus, in R (and in most statistical packages) you can get the output for an ANOVA by different routes. Let us make sure we understand this, and can interpret the output from using the different routes available to us. In fact, we have jumped from using `aov` to `lm` several times by now.

This is also a good time to recap and make sure we understand ideas we have used like "more complex models" and "number of parameters". Before we see the examples with code below, make sure we can understand how many parameters (and how many degrees of freedom are taken by the model) and what they represent in models like:

- A simple linear regression like `y ~ x`.
- A multiple linear regression like `y ~ x + z`.
- A one-way ANOVA like the one for the training regimes, with three possible training regimes: `y ~ ftraining`.
- A two-way ANOVA with interactions, like the one about Drug (two levels) and Diet (three levels): `y ~ Drug*Diet`.
- A two-way ANOVA without interactions, like one we might fit to the Drug and Diet data: `y ~ Drug + Diet`.

Note: the details about the exact numerical value of the coefficients can be skipped in a first reading. What you definitely need to understand is how many parameters we are estimating.

Let us now see a simple example:

```
asAnova <- aov(activ ~ ftraining, data = dmit)
summary(asAnova)
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## ftraining    2  31.15   15.57   22.89 1.7e-07 ***
## Residuals   43  29.26    0.68
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Not let us fit the same model using `lm`.

```
asLm <- lm(activ ~ ftraining, data = dmit)
anova(asLm)
```

```
## Analysis of Variance Table
##
## Response: activ
##           Df Sum Sq Mean Sq F value    Pr(>F)
## ftraining  2 31.147 15.5737  22.887 1.704e-07 ***
## Residuals 43 29.260  0.6805
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The ANOVA table is the same. I've used `anova` but I could have used `Anova`.

But what is this output?

```
asLm
```

```
##
```

```
## Call:
## lm(formula = activ ~ ftraining, data = dmit)
##
## Coefficients:
##        (Intercept)      ftrainingLunch  ftrainingAfternoon
##            2.10300             0.09583             1.69435
```

```
summary(asLm)
```

```
##
## Call:
## lm(formula = activ ~ ftraining, data = dmit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9030 -0.5151 -0.1642  0.5647  1.7227
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         2.10300    0.24872   8.455 1.08e-10 ***
## ftrainingLunch      0.09583    0.34434   0.278    0.782
## ftrainingAfternoon  1.69435    0.30240   5.603 1.38e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8249 on 43 degrees of freedom
## Multiple R-squared:  0.5156,Adjusted R-squared:  0.4931
## F-statistic: 22.89 on 2 and 43 DF,  p-value: 1.704e-07
```

We are being shown the fitted coefficients. They are expressed as deviations with respect to a baseline level, in this case the first level:

```
means <- with(dmit, tapply(activ, ftraining, mean)) ## instead of "tapply",
                                                    ## "by" or "aggregate"
                                                    ## would
                                                    ## also work here

means[1]
```

```
## Morning
##   2.103
```

```
means[2] - means[1]
```

```
##      Lunch
## 0.09583333
```

```
means[3] - means[1]
```

```
## Afternoon
##  1.694348
```

```
rm(means) ## let's remove it, so as not the leave
          ## garbage around
```

This is the default parameterization in R. But is is not the only one available.

Of course, the above applies to more than one factor, and to factors with an arbitrary number of

levels.

## 19.1 Changing the reference in the one-way

We can change the reference, and that will change what we are comparing against:

```
dmitb <- dmit
dmitb$ftraining2 <- factor(dmitb$ftraining,
                           levels = c("Afternoon", "Lunch", "Morning"))
## check
with(dmitb, table(ftraining, ftraining2))
```

```
##            ftraining2
## ftraining   Afternoon Lunch Morning
##   Morning           0     0      11
##   Lunch             0    12       0
##   Afternoon        23     0       0
```

Now, rerun the analysis

```
asLm2 <- lm(activ ~ ftraining2, data = dmitb)
anova(asLm2) ## no difference, of course
```

```
## Analysis of Variance Table
##
## Response: activ
##            Df Sum Sq Mean Sq F value    Pr(>F)
## ftraining2  2 31.147 15.5737  22.887 1.704e-07 ***
## Residuals  43 29.260  0.6805
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(asLm2)
```

```
##
## Call:
## lm(formula = activ ~ ftraining2, data = dmitb)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9030 -0.5151 -0.1642  0.5647  1.7227
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         3.7973     0.1720  22.077  < 2e-16 ***
## ftraining2Lunch    -1.5985     0.2938  -5.442 2.36e-06 ***
## ftraining2Morning  -1.6943     0.3024  -5.603 1.38e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8249 on 43 degrees of freedom
## Multiple R-squared:  0.5156,Adjusted R-squared:  0.4931
## F-statistic: 22.89 on 2 and 43 DF,  p-value: 1.704e-07
```

```
meansb <- with(dmitb, tapply(activ, ftraining2, mean))
```

```
meansb[1]
```

```
## Afternoon
##  3.797348
```

```
meansb[2] - meansb[1]
```

```
##     Lunch
## -1.598514
```

```
meansb[3] - meansb[1]
```

```
##   Morning
## -1.694348
```

```
rm(meansb)
```

If we go back to 16.1 we can also do the same and change the reference, and that changes what we are comparing. For instance, let us recode A and run the model again in those data:

```
df1b <- df1
df1b$A <- factor(df1b$A, levels = c("a2", "a1"))
table(df1b$A, df1$A) ## double check
```

```
##
##      a1 a2 a3
##   a2  0  2  0
##   a1  2  0  0
```

```
summary(lm(y ~ A + B, data = df1b))
```

```
##
## Call:
## lm(formula = y ~ A + B, data = df1b)
##
## Residuals:
##        1        2        4        5
## -0.09128  0.09128  0.09128 -0.09128
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.71624    0.15810   4.530    0.138
## Aa1         -0.45692    0.18255  -2.503    0.242
## Bb2         -0.02286    0.18255  -0.125    0.921
##
## Residual standard error: 0.1826 on 1 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  0.8626,Adjusted R-squared:  0.5879
## F-statistic:  3.14 on 2 and 1 DF,  p-value: 0.3706
```

## 19.2 Coefficients with two-ways

What about two-way models? Lets us first play with another fake data set:

```
y <- c(1:9, 20, 21, 22)
X <- rep(rep(c("x1", "x2"), c(3, 3)), 2)
U <- rep(c("u1", "u2"), c(6, 6))
```

```
anova(lm(y ~ U * X)) ## so strong evidence of interaction
```

```
## Analysis of Variance Table
##
## Response: y
##           Df Sum Sq Mean Sq F value    Pr(>F)
## U          1    363     363     363 5.964e-08 ***
## X          1    192     192     192 7.115e-07 ***
## U:X        1     75      75      75 2.457e-05 ***
## Residuals  8      8       1
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## The cell means
tapply(y, list(X, U), mean)
```

```
##    u1 u2
## x1  2  8
## x2  5 21
```

```
## The estimates
summary(lm(y ~ X * U))
```

```
##
## Call:
## lm(formula = y ~ X * U)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##     -1     -1      0      1      1
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.0000     0.5774   3.464  0.00852 **
## Xx2           3.0000     0.8165   3.674  0.00627 **
## Uu2           6.0000     0.8165   7.348 8.01e-05 ***
## Xx2:Uu2      10.0000     1.1547   8.660 2.46e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1 on 8 degrees of freedom
## Multiple R-squared:  0.9875,Adjusted R-squared:  0.9828
## F-statistic:   210 on 3 and 8 DF,  p-value: 6.053e-08
```

```
## The intercept is the first cell mean.
## The right and bottom cell:
## the X2:U2 =
21 - (2 + 3 + 6)
```

```
## [1] 10
```

```
## The coefficient for x2 is the difference between the intercept and
## the first column of the second row:
5 - 2
```

```
## [1] 3
```

```
## The coefficient for u2 is the difference between the intercept
## and the first row of the second column:
8 -2
```

```
## [1] 6
```

So things here are easy: a saturated model with interactions has a many parameters as cell means and once we figure out what is the reference, we can see what each coefficient means.

What about additive models? This is slightly more complicated:

```
summary(mxu <- lm(y ~ X  + U))
```

```
##
## Call:
## lm(formula = y ~ X + U)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##   -3.5   -2.5    0.0    2.5    3.5
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -0.500      1.518  -0.329 0.749468
## Xx2            8.000      1.753   4.563 0.001361 **
## Uu2           11.000      1.753   6.274 0.000145 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.037 on 9 degrees of freedom
## Multiple R-squared:  0.8699,Adjusted R-squared:  0.841
## F-statistic: 30.09 on 2 and 9 DF,  p-value: 0.0001033
```

```
model.matrix(mxu)
```

```
##    (Intercept) Xx2 Uu2
## 1            1   0   0
## 2            1   0   0
## 3            1   0   0
## 4            1   1   0
## 5            1   1   0
## 6            1   1   0
## 7            1   0   1
## 8            1   0   1
## 9            1   0   1
## 10           1   1   1
## 11           1   1   1
## 12           1   1   1
## attr(,"assign")
## [1] 0 1 2
## attr(,"contrasts")
## attr(,"contrasts")$X
## [1] "contr.treatment"
##
## attr(,"contrasts")$U
```

```
## [1] "contr.treatment"

mean(y) ## 9

## [1] 9

## What is the first cell? the overall mean with the effect of X:1 and
## U:1, which is the overall mean minus half the effects of X:2 and U:2

9 - 11/2 - 8/2 ## Where 11 and 8 are coming from the fitted model

## [1] -0.5

fitted(mxu) ## Yes: fitted for first group are the intercept term

##    1    2    3    4    5    6    7    8    9   10   11   12
## -0.5 -0.5 -0.5  7.5  7.5  7.5 10.5 10.5 10.5 18.5 18.5 18.5

## But why the 11 and the 8 above?
## Again, look here:
(mxu <- tapply(y, list(X, U), mean))

##    u1 u2
## x1  2  8
## x2  5 21

## or here
aggregate(y ~ X * U, FUN = mean)

##    X  U  y
## 1 x1 u1  2
## 2 x2 u1  5
## 3 x1 u2  8
## 4 x2 u2 21

## And our model says: an overall mean, and row and column deviations.
## Let's write it.
## Row effects, or effect of X:2 (effect of X:1 = - effect of X:2)
## is the average of the row differences at each column
## or average of (5 - 2) and (21 - 8) which is the effect of X:2
0.5 * ((5 - 2) + (21 - 8)) # = 8

## [1] 8

## or, similarly
mean(mxu[2, ] - mxu[1, ])

## [1] 8

## Similar for column effects, or the effect of U:2
## effect of U:2:
0.5 * ((8 - 2) + (21 - 5)) # = 11

## [1] 11

## or, similarly
mean(mxu[ , 2] - mxu[, 1])

## [1] 11
```

```
## So the first cell is the first cell UNDER that additive model. You can't
## just put the first cell mean in there.
```

What if we go to the data in section 16.1? Again, in the interaction case things are easiest:

```
(means <- with(df1, tapply(y, list(A, B), mean)))
```

```
##           b1        b2
## a1 0.1680415 0.3277343
## a2 0.8075164 0.6021007
## a3 0.3849424 0.6043941
```

```
summary(m2 <- lm(y ~ A * B, data = df1))
```

```
##
## Call:
## lm(formula = y ~ A * B, data = df1)
##
## Residuals:
## ALL 6 residuals are 0: no residual degrees of freedom!
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.16804         NA      NA       NA
## Aa2          0.63947         NA      NA       NA
## Aa3          0.21690         NA      NA       NA
## Bb2          0.15969         NA      NA       NA
## Aa2:Bb2     -0.36511         NA      NA       NA
## Aa3:Bb2      0.05976         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
## Multiple R-squared:      1,Adjusted R-squared:    NaN
## F-statistic:   NaN on 5 and 0 DF,  p-value: NA
```

```
## each main effect
means[1, 2] - means[1, 1]
```

```
## [1] 0.1596928
```

```
means[2, 1] - means[1, 1]
```

```
## [1] 0.6394749
```

```
## interaction
means[2, 2] - ( means[1, 1] + (m2$coefficients[2] + m2$coefficients[3]))
```

```
##        Aa2
## -0.4223165
```

What about additive model?

```
## Overall mean:
mean(df1$y)
```

```
## [1] 0.4824549
```

```
## Note what are the estimates of the effects of A and B: the mean deviation
## of the second level from the first:
## A2
```

```
mean(means[2, ] - means[1, ])
```

```
## [1] 0.4569206
```

```
## B2
mean(means[, 2] - means[, 1])
```

```
## [1] 0.05790959
```

```
## Intercept
mean(df1$y) -
    0.5 * (mean(means[2, ] - means[1, ])) -
    0.5 * mean(means[, 2] - means[, 1])
```

```
## [1] 0.2250398
```

## 19.3 Other contrasts

We are using `contr.treatment`, the default in R. There are other types. In particular, and when we want to estimate effects in the presence of interactions, we probably want to use `contr.sum`. We will not pursue this any further here.

To give you a quick taste, this might do:

(We will use `contr.Sum` from *car*, since clearer labeling)

```
opt <- options(contrasts = c("contr.Sum", "contr.poly"))
m11 <- lm(y ~ A + B, data = df1)
anova(m11)
```

```
## Analysis of Variance Table
##
## Response: y
##           Df   Sum Sq  Mean Sq F value Pr(>F)
## A          2 0.209224 0.104612  3.9552 0.2018
## B          1 0.005030 0.005030  0.1902 0.7053
## Residuals  2 0.052898 0.026449
```

```
summary(m11)
```

```
##
## Call:
## lm(formula = y ~ A + B, data = df1)
##
## Residuals:
##        1        2        3        4        5        6
## -0.05089  0.13166 -0.08077  0.05089 -0.13166  0.08077
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.48245    0.06639   7.267   0.0184 *
## A[S.a1]     -0.23457    0.09390  -2.498   0.1298
## A[S.a2]      0.22235    0.09390   2.368   0.1414
## B[S.b1]     -0.02895    0.06639  -0.436   0.7053
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.1626 on 2 degrees of freedom
## Multiple R-squared:  0.802,Adjusted R-squared:  0.505
## F-statistic:   2.7 on 3 and 2 DF,  p-value: 0.2818
```

```
(overallMean <- mean(df1$y))
```

```
## [1] 0.4824549
```

```
mA <- with(df1, tapply(y, A, mean))
mA - overallMean
```

```
##          a1          a2          a3
## -0.23456697  0.22235365  0.01221332
```

```
mB <- with(df1, tapply(y, B, mean))
mB - overallMean
```

```
##          b1          b2
## -0.02895479  0.02895479
```

Interaction now (note the estimates!)

```
m12 <- lm(y ~ A * B, data = df1)
anova(m12)
```

```
## Warning in anova.lm(m12):  ANOVA F-tests on an essentially perfect fit are unreliable
```

```
## Analysis of Variance Table
##
## Response: y
##           Df   Sum Sq  Mean Sq F value Pr(>F)
## A          2 0.209224 0.104612
## B          1 0.005030 0.005030
## A:B        2 0.052898 0.026449
## Residuals  0 0.000000
```

```
summary(m12)
```

```
##
## Call:
## lm(formula = y ~ A * B, data = df1)
##
## Residuals:
## ALL 6 residuals are 0: no residual degrees of freedom!
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      0.48245         NA      NA       NA
## A[S.a1]         -0.23457         NA      NA       NA
## A[S.a2]          0.22235         NA      NA       NA
## B[S.b1]         -0.02895         NA      NA       NA
## A[S.a1]:B[S.b1] -0.05089         NA      NA       NA
## A[S.a2]:B[S.b1]  0.13166         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
## Multiple R-squared:      1,Adjusted R-squared:    NaN
```

```
## F-statistic:   NaN on 5 and 0 DF,  p-value: NA
```

What if we changed the reference?

```
summary(lm(y ~ A + B, data = df1b))
```

```
##
## Call:
## lm(formula = y ~ A + B, data = df1b)
##
## Residuals:
##        1        2        4        5
## -0.09128  0.09128  0.09128 -0.09128
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.47635    0.09128   5.219    0.121
## A[S.a2]      0.22846    0.09128   2.503    0.242
## B[S.b1]      0.01143    0.09128   0.125    0.921
##
## Residual standard error: 0.1826 on 1 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  0.8626,Adjusted R-squared:  0.5879
## F-statistic:  3.14 on 2 and 1 DF,  p-value: 0.3706
```

Return contrasts to usual state

```
options(opt)
```

## 19.4 Changing the reference in two-ways

If we change the levels or references, of course the interpretation of the coefficients must also change. This is nothing new relative to what we saw for the one-way.

## 19.5 Unbalanced case

These examples have used balanced data. Things get trickier with unbalanced data. The idea of this section is to get an intuitive understanding of what the numbers mean, but then for real you'll do this with a computer.

## 19.6 Three way anovas, factors with more than two classes, etc

There is nothing conceptually new, but the accounting gets more complicated.

# 20 Continuous and discrete independent variables and ANCOVA

Right now, nothing should stop us from thinking about models where the right hand side contains both continuous and discrete variables. Let's do it with the cystic fibrosis data set. ANCOVA refers to this mixture of ANOVA and regression and stands for "Analysis of covariance"[21]. Regardless, all of ANOVA, regression, and ANCOVA are especial types of linear models.

Let's fit a model where the independent variables are sex (discrete, obviously) and age. However, sex is coded with 0/1 and we want it to be a factor, explictly. Let's recode it:

```
cystfibr2$sex <- factor(cystfibr2$sex, labels = c('Male','Female'))
```

```
mcyst2 <- lm(pemax ~ age * sex, data = cystfibr2)
summary(mcyst2)
```

```
##
## Call:
## lm(formula = pemax ~ age * sex, data = cystfibr2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -54.901 -12.447   5.069  15.099  45.099
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)     54.185     20.806   2.604  0.01656 *
## age              4.162      1.281   3.249  0.00384 **
## sexFemale        5.683     37.968   0.150  0.88243
## age:sexFemale   -1.313      2.602  -0.505  0.61911
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.25 on 21 degrees of freedom
## Multiple R-squared:  0.419,Adjusted R-squared:  0.336
## F-statistic: 5.048 on 3 and 21 DF,  p-value: 0.008655
```

```
Anova(mcyst2)
```

```
## Anova Table (Type II tests)
##
## Response: pemax
##            Sum Sq Df F value   Pr(>F)
## age        8819.5  1 11.8802 0.002417 **
## sex         955.4  1  1.2870 0.269386
## age:sex     189.0  1  0.2546 0.619111
## Residuals 15589.7 21
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

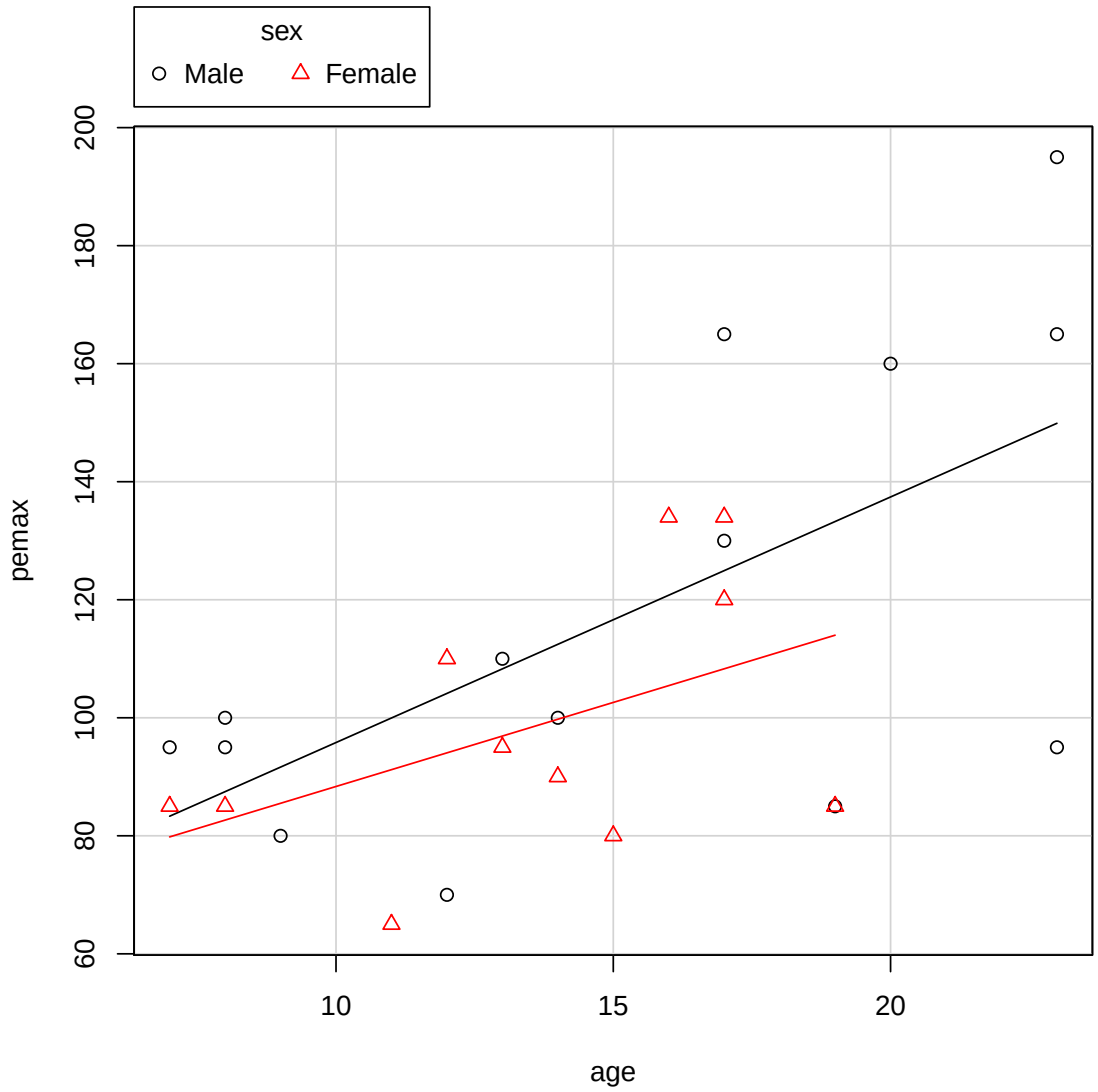
Note we added a "*", so an interaction between a continuous and a discrete variable. Here there

[21] But this does not mean that we are comparing covariances, as in comparing correlations, between groups; we are comparing groups after adjusting for possible covariates, if that is warranted by the absence of interactions between the continuous and discrete predictors.

is no evidence of interaction. But, what would an interaction have looked like? Different slopes for each group.

Look at the plots:

```
scatterplot(pemax ~ age | sex, smooth = FALSE, data = cystfibr2)
```




(yes, the best fitting slopes are slightly different, but they are not significantly different, as shown by the the "age:sex[T.Female]" term in the model above, so no evidence for different slopes).

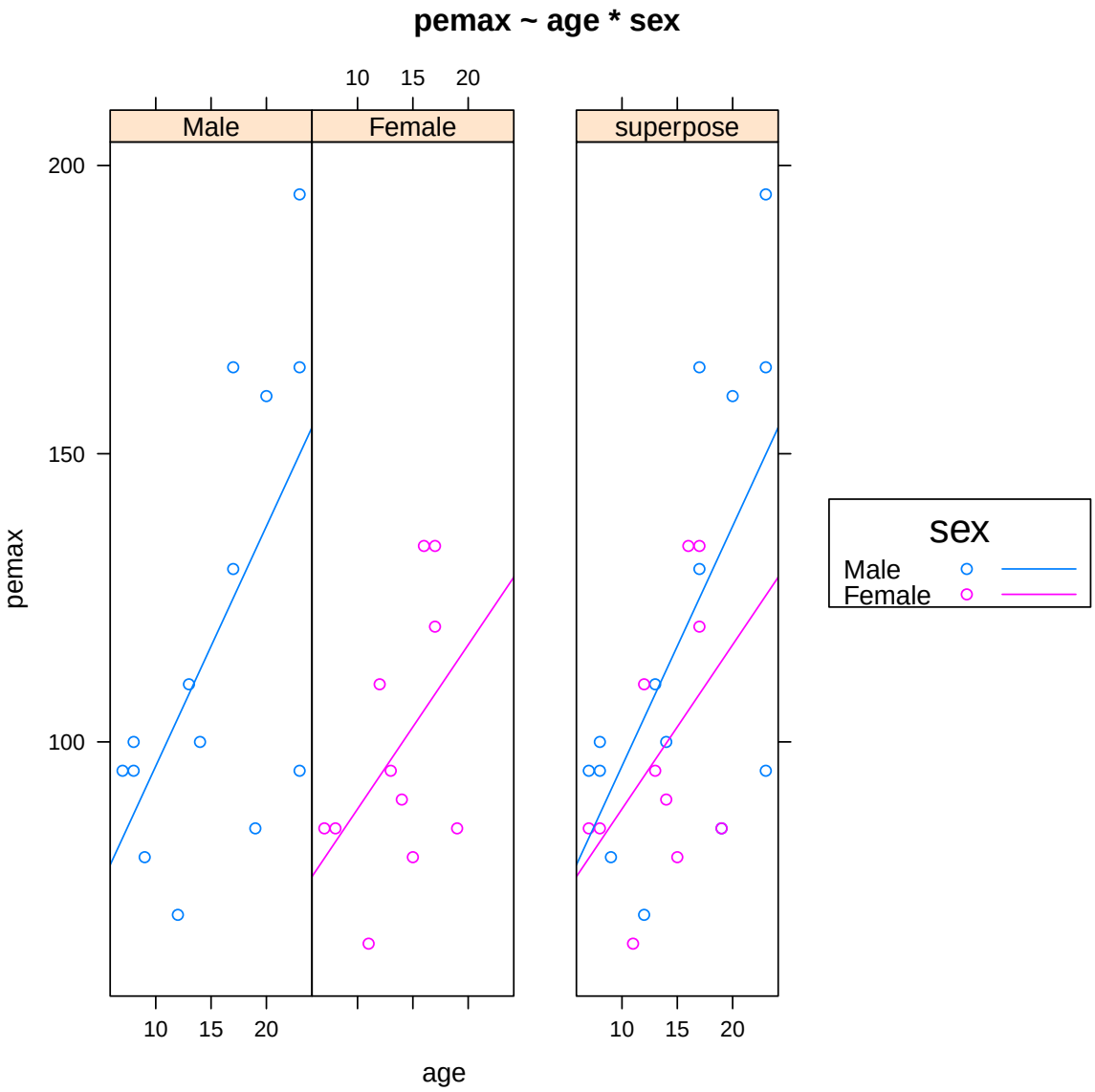
The different intercepts are captured by the term "sex[T.Female]" (that is not significant in this example either). Anyway, we can often have models where we have no evidence of different slopes (no interaction), but evidence of different intercepts: these means parallel lines (we will see one below: 20.3).

One can visualize this also with the function `ancova` in package *HH*, so we do not need to load it again); `ancova` also produces and anova table (with sequential sums of squares, Type I):

```
ancova(pemax ~ age * sex, data = cystfibr2)
```

```
## Analysis of Variance Table
##
## Response: pemax
```

```
##           Df  Sum Sq Mean Sq F value   Pr(>F)
## age        1 10098.5 10098.5 13.6031 0.001366 **
## sex        1   955.4   955.4  1.2870 0.269386
## age:sex    1   189.0   189.0  0.2546 0.619111
## Residuals 21 15589.7   742.4
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```




Note that we are using `ancova` just to produce plots. The analysis are directly available using `lm`, `anova`, etc.

## 20.1 A parallel slopes model

We could have started by fitting a model with parallel slopes:

```
mcyst0 <- lm(pemax ~ age + sex, data = cystfibr2)
summary(mcyst0)

##
```

```
## Call:
## lm(formula = pemax ~ age + sex, data = cystfibr2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.423 -13.617   5.637  17.485  47.577
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   59.027     18.146   3.253  0.00365 **
## age            3.843      1.096   3.507  0.00199 **
## sexFemale    -12.632     10.944  -1.154  0.26081
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 26.78 on 22 degrees of freedom
## Multiple R-squared:  0.412,Adjusted R-squared:  0.3585
## F-statistic: 7.706 on 2 and 22 DF,  p-value: 0.002907
```

```
Anova(mcyst0)
```

```
## Anova Table (Type II tests)
##
## Response: pemax
##            Sum Sq Df F value   Pr(>F)
## age        8819.5  1 12.2969 0.001992 **
## sex         955.4  1  1.3321 0.260810
## Residuals 15778.7 22
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Make sure you understand the differences with respect to the previous model. What are we fitting here? What are we saying, biologically, in this model?

## 20.2 Formally comparing models

In this case, the sequential anova table (as produced by `ancova` or `anova`) suggests that we could simplify our model a lot, and use one with a single intercept and slope (i.e., just like a simple linear regression as in section 17) as neither "sex" by itself nor the interaction is relevant. We will do that, and we will then do a global model comparison to verify the simplified model is a reasonable one:

```
mcyst3 <- lm(pemax ~ age, data = cystfibr2)
anova(mcyst3, mcyst2)
```

```
## Analysis of Variance Table
##
## Model 1: pemax ~ age
## Model 2: pemax ~ age * sex
##   Res.Df   RSS Df Sum of Sq      F Pr(>F)
## 1     23 16734
## 2     21 15590  2    1144.4 0.7708 0.4753
```

This is a comparison of two models using an F test: it tests whether the larger (more complex,

with more terms) model is significantly better than the smaller one. It clearly shows that, in this case, that the larger model model (the one with both a main effect of "sex" and an interaction) is not significantly better than the one without "sex". So we can just keep model `mcyst3`: there is no statistical evidence of `mcyst2` being any better.

Three comments here:

- These tests only make sense for nested models (where the terms of one of the models is a subset of terms of the other). Beware that R does not check this, and you could easily do meaningless things[22].
- Whether you type `anova(mcyst2, mcyst3)` or `anova(mcyst3, mcyst2)` is inconsequential for the F statistic and p-values.
- This was a very clear-cut case. Often, people will proceed in steps: first check no interaction and then, later, and if no interaction, check if there is a need for different intercepts.
  In fact, we could have done this in a more step-by-step way:

```
anova(mcyst0, mcyst2)
## Analysis of Variance Table
##
## Model 1: pemax ~ age + sex
## Model 2: pemax ~ age * sex
##   Res.Df   RSS Df Sum of Sq      F Pr(>F)
## 1     22 15779
## 2     21 15590  1       189 0.2546 0.6191
```

  where we compare the model and without interaction (though this is the same, of course, as the term for interaction in `Anova(mcyst2)`).
  And then

```
anova(mcyst3, mcyst0)
## Analysis of Variance Table
##
## Model 1: pemax ~ age
## Model 2: pemax ~ age + sex
##   Res.Df   RSS Df Sum of Sq      F Pr(>F)
## 1     23 16734
## 2     22 15779  1    955.43 1.3321 0.2608
```

  but, again, this is just the same as the term for `sex` in `Anova(mcyst0)`.

## 20.3 ANCOVA with the birds and the reptiles

We will use the longevity and metabolic rate data we used in section 17, but now the full one: `anage_a_r`. We will go pretty fast here (this is just a kind of review), and will examine two things:

- If the relationship between metabolic rate and body mass is different between reptiles and birds.
- If the relationship between longevity and body mass is different between reptiles and birds.

We will see interactions, parallel and non-parallel slopes, and more comparison of models. Again, these analyses are not fully correct as we ignore phylogenetic relatedness. But they are nice to illustrate a couple of points. We will directly use log transformations (again, theory and previous empirical evidence indicate this is the way to go —and I looked at a couple of different models already).

First, metabolic rate vs. body mass allowing for interaction with "Class" (bird vs. reptile):

[22] There are ways to compare non-nested models using other procedures, for instance based on AIC.

```
metab_b_r <- lm(logMetabolicRate ~ logBodyMass * Class, data = anage_a_r)
summary(metab_b_r)
```

```
##
## Call:
## lm(formula = logMetabolicRate ~ logBodyMass * Class, data = anage_a_r)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.26958 -0.14488  0.01647  0.18083  0.62902
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -3.15949    0.05366  -58.88   <2e-16 ***
## logBodyMass                 0.65037    0.01201   54.17   <2e-16 ***
## ClassReptilia              -2.93984    0.25722  -11.43   <2e-16 ***
## logBodyMass:ClassReptilia -0.04577    0.04488   -1.02    0.309
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2688 on 174 degrees of freedom
##   (1548 observations deleted due to missingness)
## Multiple R-squared:  0.9576,Adjusted R-squared:  0.9569
## F-statistic:  1310 on 3 and 174 DF,  p-value: < 2.2e-16
```

The output is clear: parallel lines (i.e., different intercepts, but same slope). Note that this is not a silly or irrelevant biological detail: metabolic rates scales with body mass in the same way in an endothermic group (birds) and a ectothermic one (reptiles), but their metabolic rates are not the same (birds' are higher).

We could simplify this model to a model without the interaction (so single slope, two intercepts):

```
metab_b_r_2 <- lm(logMetabolicRate ~ logBodyMass + Class, data = anage_a_r)
summary(metab_b_r_2)
```

```
##
## Call:
## lm(formula = logMetabolicRate ~ logBodyMass + Class, data = anage_a_r)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.28901 -0.14756  0.01835  0.18542  0.63129
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -3.14600    0.05201  -60.49   <2e-16 ***
## logBodyMass    0.64709    0.01157   55.93   <2e-16 ***
## ClassReptilia -3.18852    0.08201  -38.88   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2688 on 175 degrees of freedom
##   (1548 observations deleted due to missingness)
## Multiple R-squared:  0.9573,Adjusted R-squared:  0.9569
```

```
## F-statistic:  1964 on 2 and 175 DF,  p-value: < 2.2e-16
```

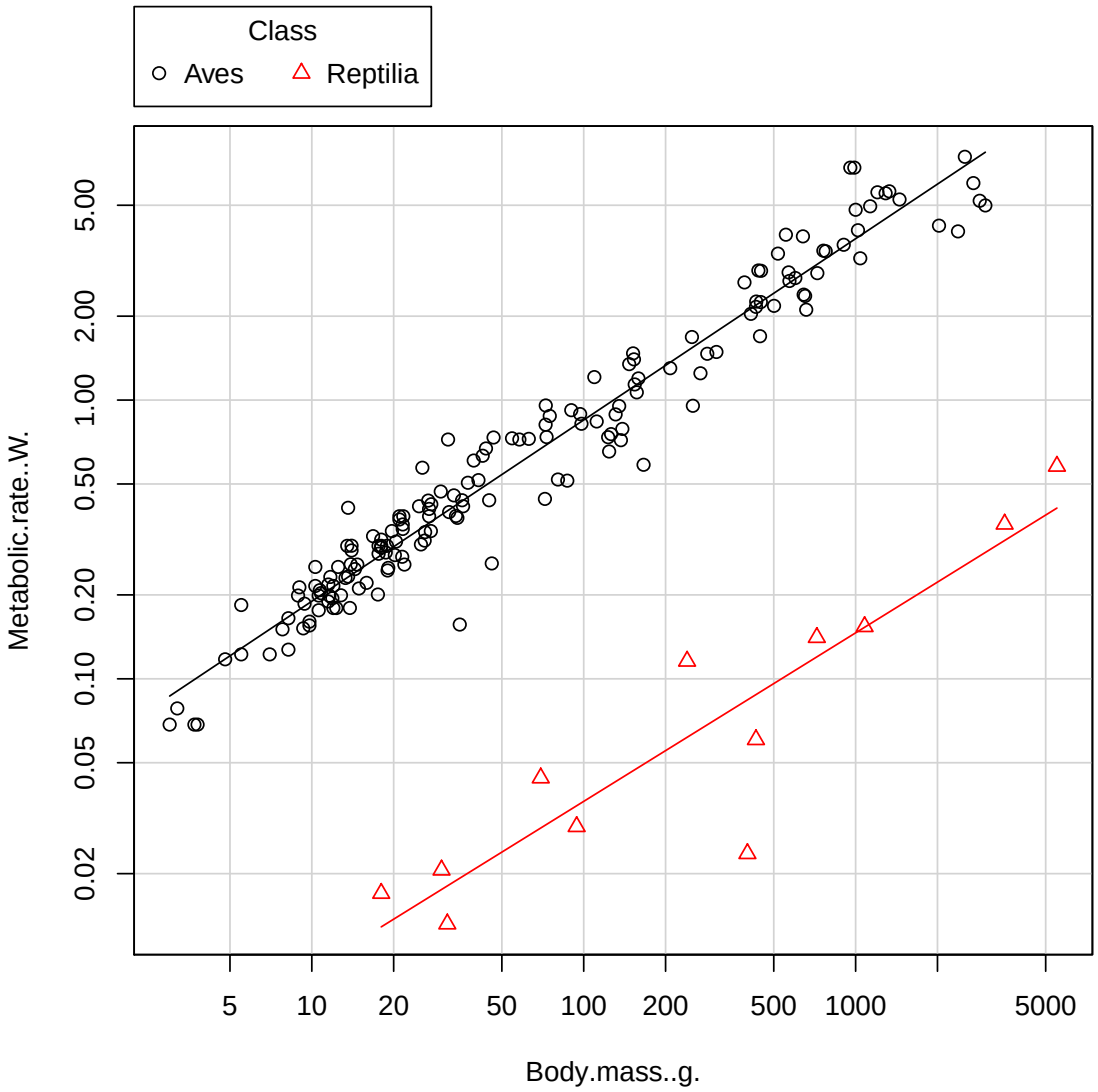
```
anova(metab_b_r_2, metab_b_r)
```

```
## Analysis of Variance Table
##
## Model 1: logMetabolicRate ~ logBodyMass + Class
## Model 2: logMetabolicRate ~ logBodyMass * Class
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1    175 12.646
## 2    174 12.571  1  0.075167 1.0404 0.3091
```

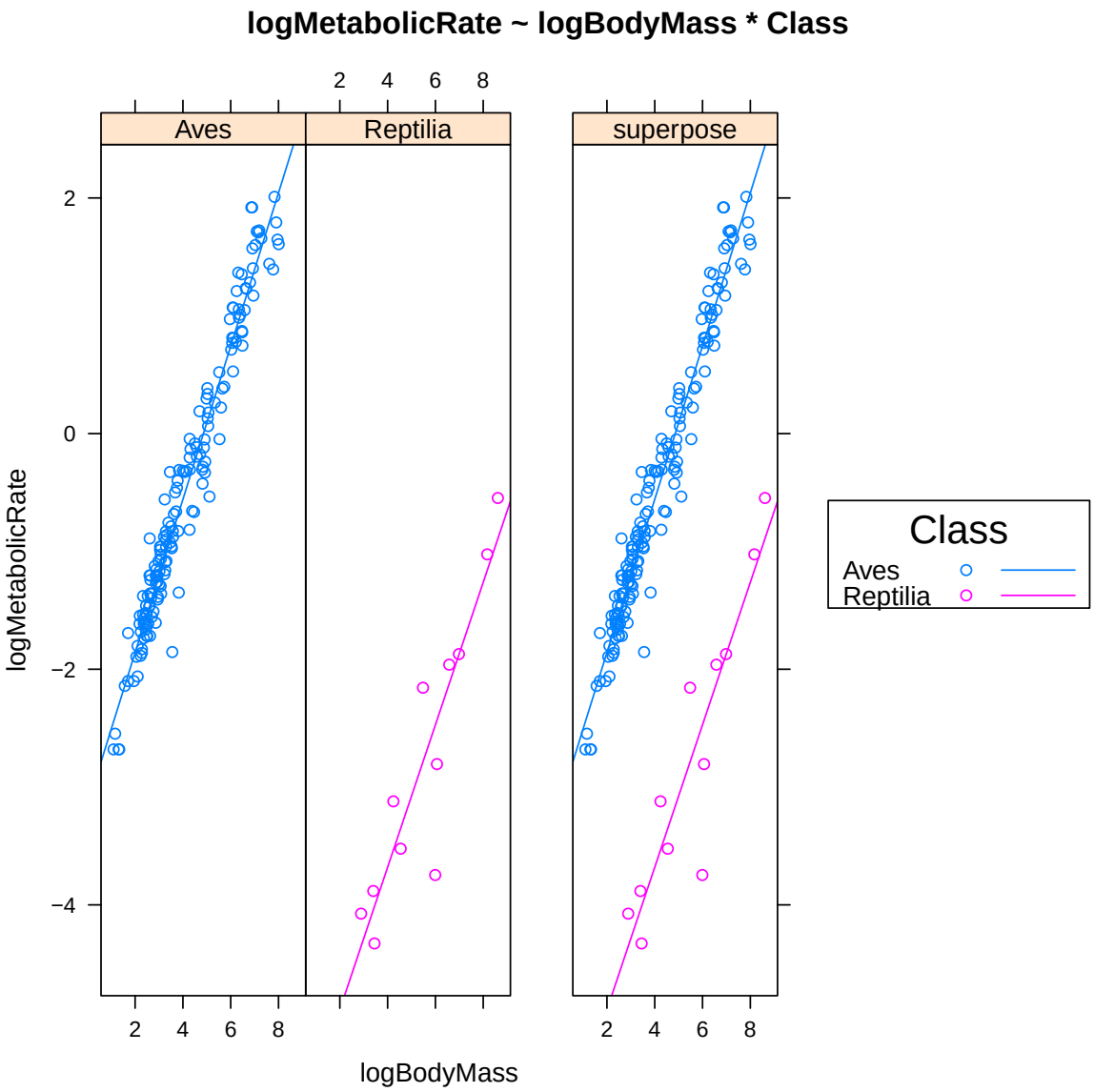
(of course, the model comparison via `anova` here is really unneeded: we know what the p-value and F values should be, since we are only removing the interaction, for which we already saw a test).

Let's see the plots:

```
scatterplot(Metabolic.rate..W. ~ Body.mass..g. | Class,
            log = "xy", smooth = FALSE, data = anage_a_r)
```



```
ancova(logMetabolicRate ~ logBodyMass * Class,
                  data = anage_a_r)
```



What about longevity?

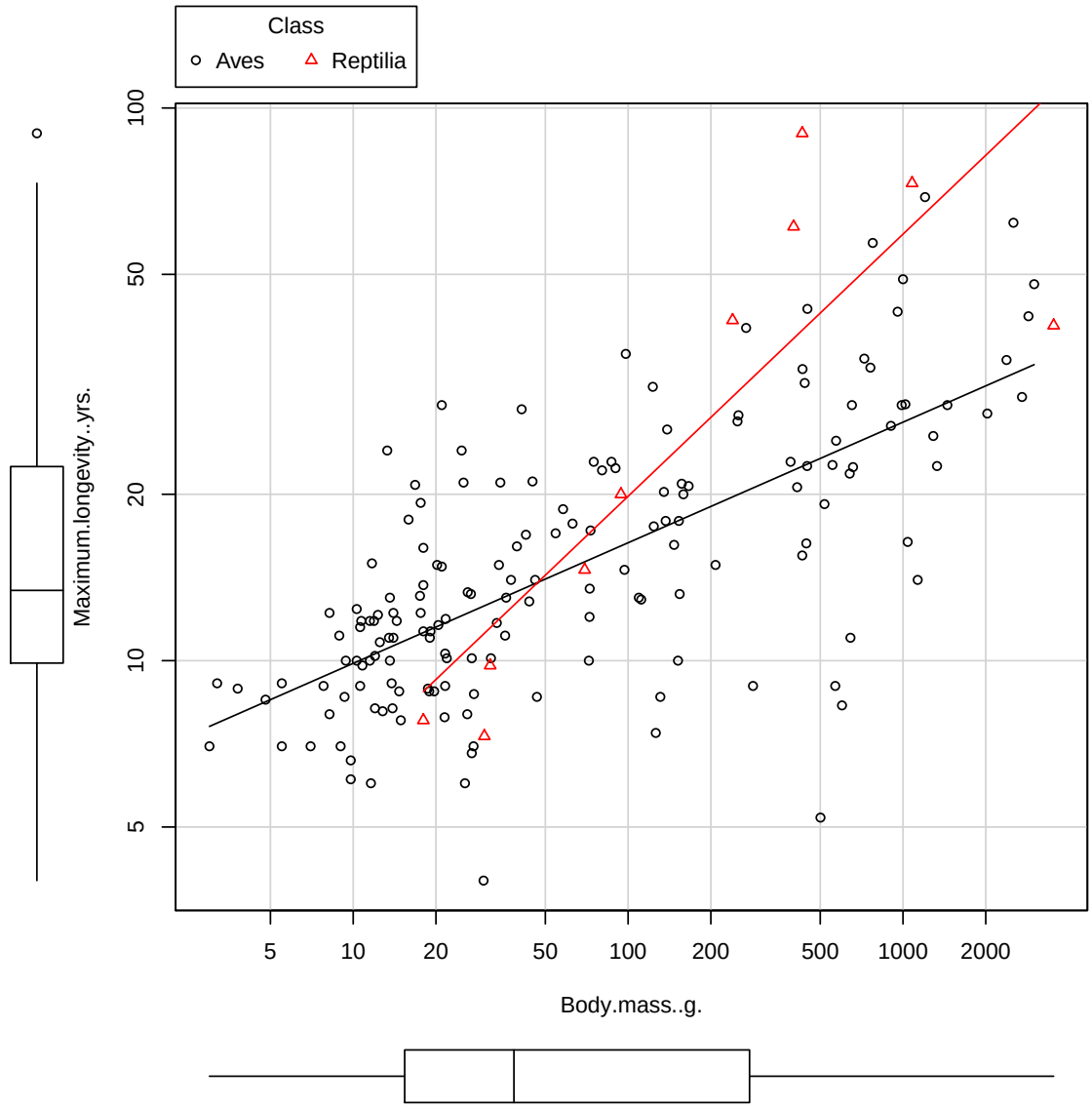
```
longev_b_r <- lm(logLongevity ~ logBodyMass * Class, data = anage_a_r)
summary(longev_b_r)
```

```
##
## Call:
## lm(formula = logLongevity ~ logBodyMass * Class, data = anage_a_r)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.49667 -0.21790  0.01212  0.20800  0.91414
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 1.78882    0.08159  21.923  < 2e-16 ***
## logBodyMass                 0.21821    0.01820  11.991  < 2e-16 ***
## ClassReptilia              -0.98582    0.42928  -2.296  0.02289 *
## logBodyMass:ClassReptilia  0.25611    0.08052   3.181  0.00175 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4028 on 168 degrees of freedom
##   (1554 observations deleted due to missingness)
## Multiple R-squared:  0.5402,Adjusted R-squared:  0.532
## F-statistic:  65.8 on 3 and 168 DF,  p-value: < 2.2e-16
```

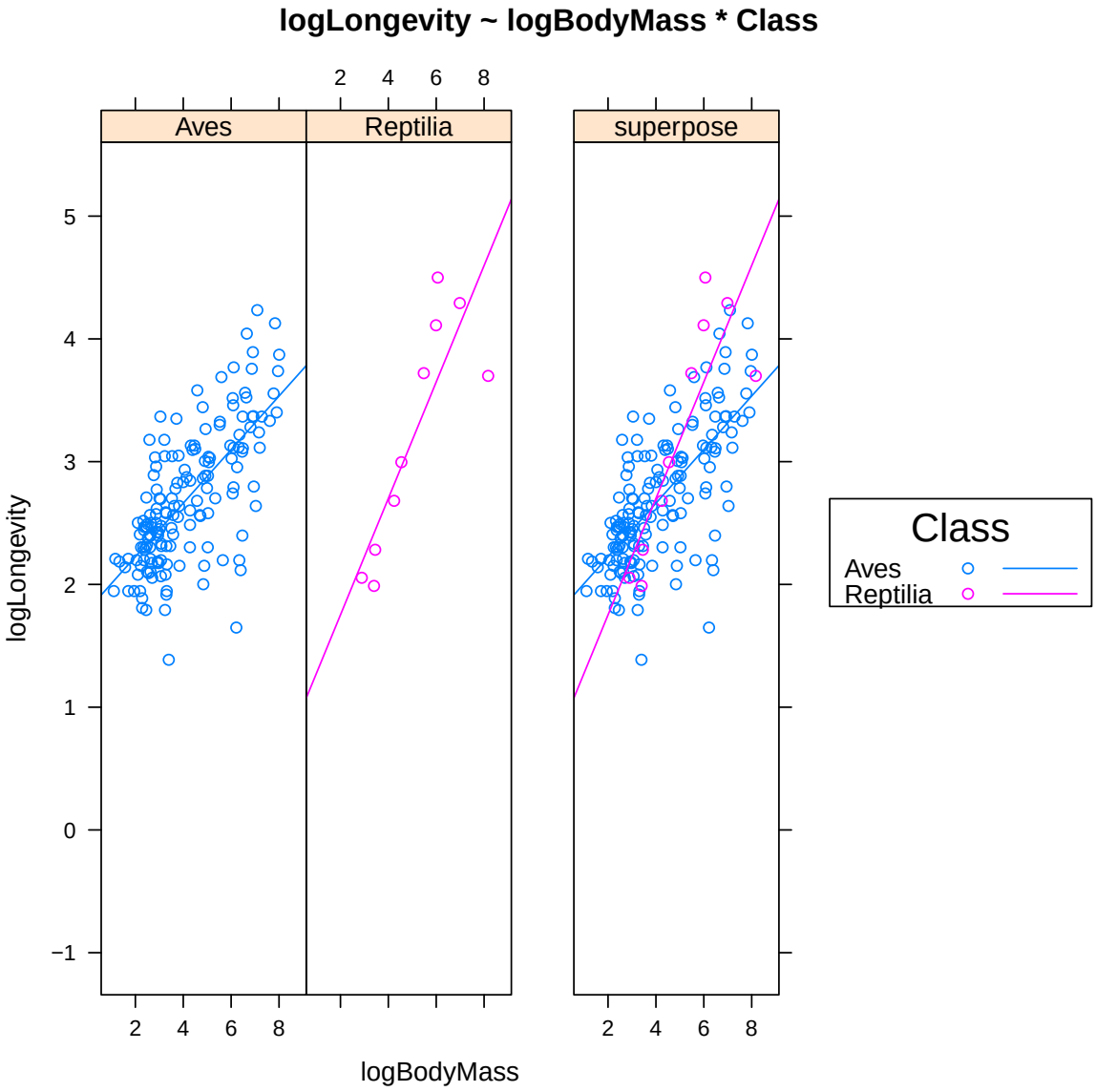
In this case, lines are not parallel: the rate of change of longevity with body mass is faster in reptiles than in birds (note the coefficient "logBodyMass:Class[T.Reptilia]").

How do things look?

```
scatterplot(Maximum.longevity..yrs. ~ Body.mass..g. | Class,
            smooth = FALSE, data = anage_a_r,
            log = "xy", boxplots = 'xy')
```




```
ancova(logLongevity ~ logBodyMass * Class, data = anage_a_r)
```




Note: the tightness of the data around the lines in this model for longevity is not nearly as good as for metabolic rate, which probably does make biological sense.

If we where we to remove "Class" and refit a simpler model we would see that the larger model is clearly better when using `anova` to compare the two models:

```
longev_b_r_2 <- lm(logLongevity ~ logBodyMass,  data = anage_a_r)
anova(longev_b_r_2, longev_b_r)
```

```
## Analysis of Variance Table
##
## Model 1: logLongevity ~ logBodyMass
## Model 2: logLongevity ~ logBodyMass * Class
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1    170 29.806
## 2    168 27.258  2    2.5477 7.8512 0.00055 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

### 20.4 More examples

Section 12.7 of Dalgaard's "Introductory statistics with R" contains a beautiful and detailed example of ANCOVA, including transformation of variables, using the `hellung` dataset included in *ISwR*.

### 20.5 More variables

We can extend the models to incorporate more variables, add interactions, etc. Interactions can involve more than two variables, can involve continuous and discrete variables, etc.

### 20.6 Parameters, coefficients

There is nothing conceptually new here. Go back to section 19 and make sure things make sense with these new models: how we interpret coefficients and how many parameters we have.

## 21 Dealing with ratios

In biology (as well as in other disciplines) it is common for researchers to use ratios of variables to try to standardize/normalize a variable. This procedure looks deceptively simple, but it is not. Here, we will explore some of the problems[23]. You can find more details[24] in this commentary from Curran-Everett: http://ajpadvan.physiology.org/cgi/doi/10.1152/advan.00053.2013.

Before we start, though, note that this section should not really be necessary after previous sections :-).

So that nothing is hidden, I will simulate the data here, but in these notes I won't provide details about how/why the data have been simulated that way (the code is below; look at it if you want). We will pretend there is a response variable, called "Y", two groups ("g1" and "g2") and another variable, called "Z". Z might be some reporter protein, or something that can be taken as a proxy for cell volume, etc.

```
set.seed(1) ## irrelevant, but so that we all
            ## get the same numbers
n <- 20
sd <- 0.5
```

### 21.1 A misleading case with parallel lines

```
z1 <- runif(n, 1, 10)
t1 <- factor(rep(c("g1", "g2"), rep(n, 2)))
y1 <-  2 * z1 + 3 * as.numeric(t1) + rnorm(n, 0, sd)
data1 <- data.frame(Y = y1, Z = z1, Group = t1, Ratio = y1/z1)
```

```
par(mfrow = c(1, 2))
with(data1, plot(Ratio ~ Group))
with(data1, plot(Y ~ Z, col = c("red", "blue")[Group]))
```

[23] I thank Alba Concepción, a former student of BM-1, for asking me questions that prompted me to write this section. She also provided the link to Curran-Everett's paper

[24] With discussion about errors in the X variable, an issue we have not covered here

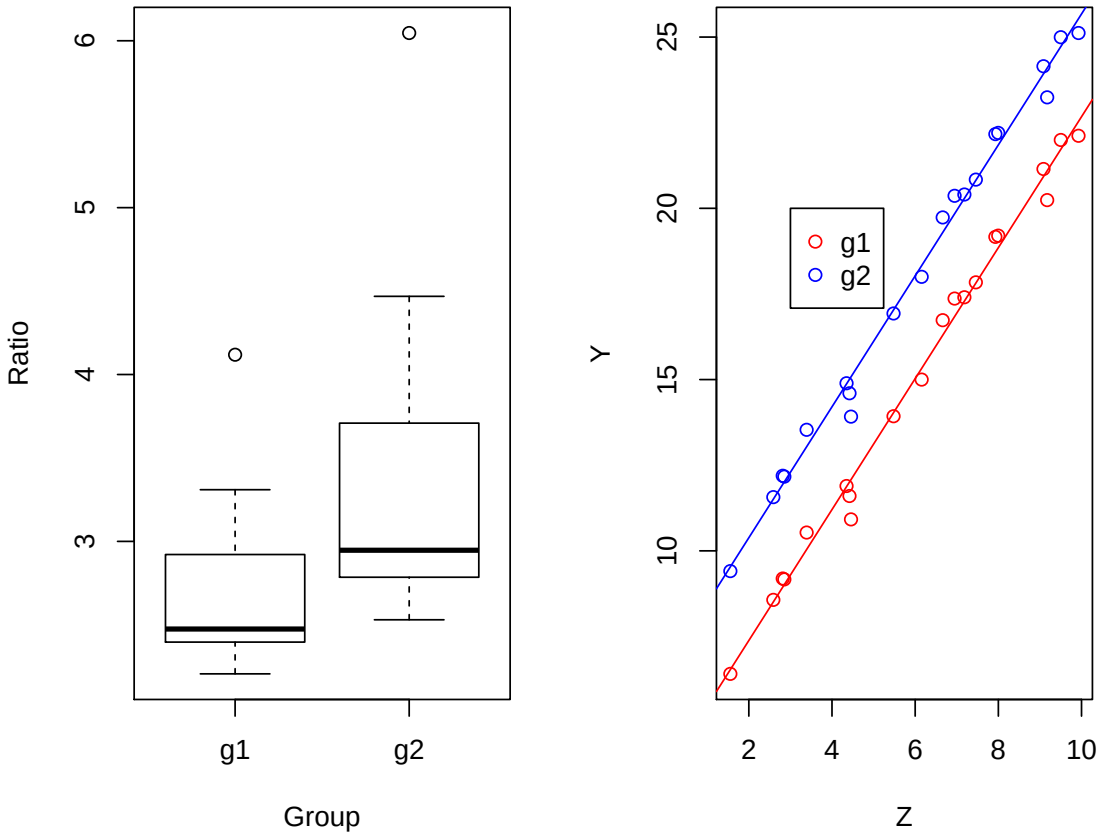



**Figure 2** – Parallel slopes but ratio differences?

```
abline(lm(Y ~ Z, subset(data1, Group == "g1")), col = "red")
abline(lm(Y ~ Z, subset(data1, Group == "g2")), col = "blue")
legend(x = 3, y = 20, legend = c("g1", "g2"), col = c("red", "blue"),
       pch = 1)
```

```
summary(lm(Y ~ Z * Group, data = data1))
```

```
##
## Call:
## lm(formula = Y ~ Z * Group, data = data1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.15961 -0.17470  0.08719  0.29409  0.52719
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 3.559e+00  2.670e-01  13.329 1.72e-15 ***
## Z           1.912e+00  4.108e-02  46.538  < 2e-16 ***
## Groupg2     3.000e+00  3.776e-01   7.945 1.97e-09 ***
## Z:Groupg2   4.146e-16  5.809e-02   0.000        1
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.461 on 36 degrees of freedom
## Multiple R-squared:  0.9925,Adjusted R-squared:  0.9919
## F-statistic:  1585 on 3 and 36 DF,  p-value: < 2.2e-16
```

```
t.test(Ratio ~ Group, data = data1)
```

```
##
##  Welch Two Sample t-test
##
## data:  Ratio by Group
## t = -2.8577, df = 29.435, p-value = 0.007759
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.1059664 -0.1836097
## sample estimates:
## mean in group g1 mean in group g2
##         2.676243         3.321031
```

```
## This is the equivalent to the t-test, except the
## t-test used by default by R does not assume equal variances
summary(aov(Ratio ~ Group, data = data1))
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## Group        1  4.158   4.158   8.166 0.00689 **
## Residuals   38 19.346   0.509
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The lines are perfectly parallel, but the test for ratios says they differ. Why? Because it is forcing a regression through the origin. Note the the linear model does get the results right: there are no differences in the rate of change of Y relative to Z, but the groups differ in intercept. As you can see, the analysis with ratios is misleading.

## 21.2 A misleading case where ratios differ

Generate the data:

```
set.seed(123)
sd <- 0.1
z2 <- seq(from = 1, to = 3, length.out = n)
ya <- z2 + rnorm(n, 0, sd)
yb <- 0.5 * z2 + 1 + rnorm(n, 0, sd)
y <- c(ya, yb)
tf <- factor(rep(c("g1", "g2"), rep(n, 2)))
z <- rep(z2, 2)
data2 <- data.frame(Y = y, Z = z, Group = tf, Ratio = y/z)
```

The figure and analysis:

```
par(mfrow = c(1, 2))
with(data2, plot(Ratio ~ Group))
with(data2, plot(Y ~ Z, col = c("red", "blue")[Group]))
abline(lm(Y ~ Z, subset(data2, Group == "g1")), col = "red")
```

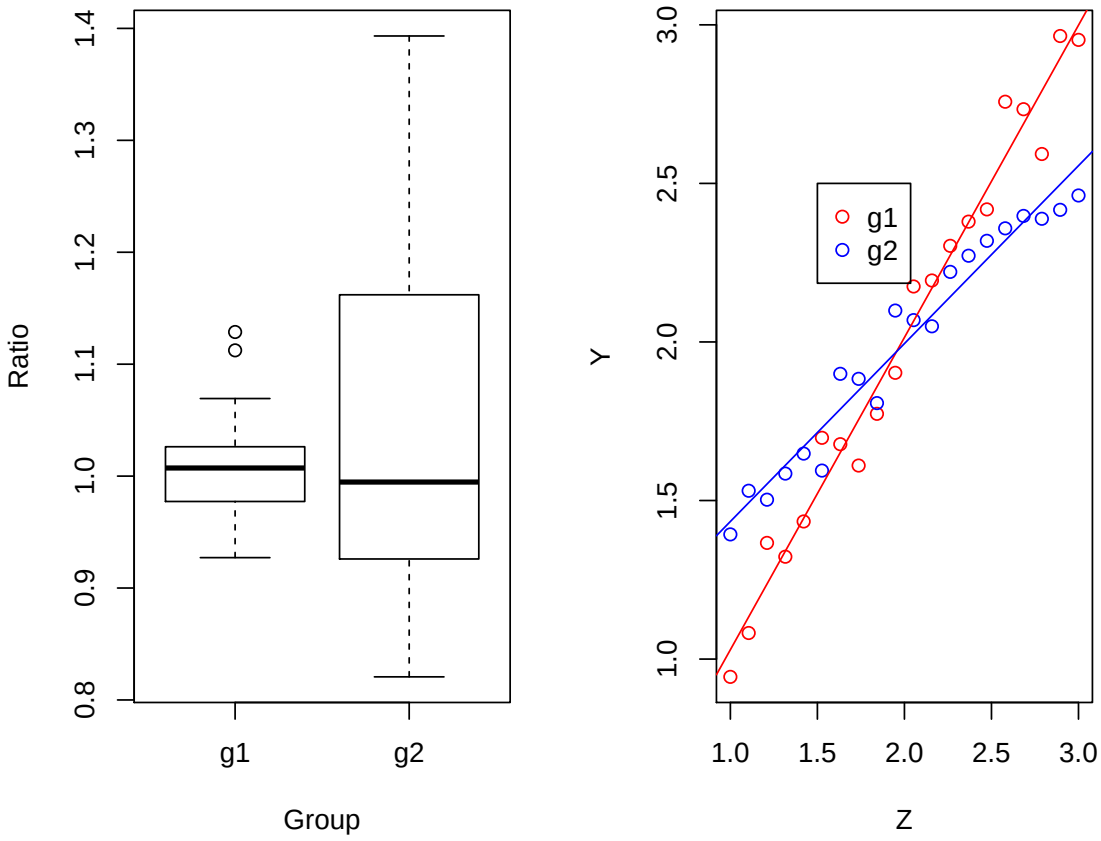



**Figure 3** – Different slopes and no ratio differences?

```
abline(lm(Y ~ Z, subset(data2, Group == "g2")), col = "blue")
legend(x = 1.5, y = 2.5, legend = c("g1", "g2"), col = c("red", "blue"),
       pch = 1)
```

```
summary(lm(Y ~ Z * Group, data = data2))
```

```
##
## Call:
## lm(formula = Y ~ Z * Group, data = data2)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.198791 -0.052996 -0.003768  0.049134  0.173353
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.04465    0.06808   0.656    0.516
## Z            0.98476    0.03258  30.230  < 2e-16 ***
## Groupg2      0.82820    0.09629   8.601 2.96e-10 ***
## Z:Groupg2   -0.42374    0.04607  -9.198 5.52e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.08843 on 36 degrees of freedom
## Multiple R-squared:  0.9711,Adjusted R-squared:  0.9687
## F-statistic: 403.6 on 3 and 36 DF,  p-value: < 2.2e-16
```

```
t.test(Ratio ~ Group, data = data2) ## or do an ANOVA, as above
```

```
##
##  Welch Two Sample t-test
##
## data:  Ratio by Group
## t = -0.90546, df = 22.839, p-value = 0.3747
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.11757840  0.04600486
## sample estimates:
## mean in group g1 mean in group g2
##         1.008799         1.044586
```

The ratios do differ, if we do not force them through the origin: the rate of change of Y relative to Z differs between the two groups. But just comparing the ratios will not detect it. Again, a linear model does detect it just fine: you see a strong interaction between slope and group. And, again, the analysis of ratios is misleading.

### 21.3 Diagnostics et al. and other warning signs

If you look closely, the data show warning signals, such as possible differences in variances of ratios. And if you have no idea about the model, you might be able to compare several and use diagnostics, as explained in section 23 to choose among models. But, in general, ratios are not the way to. And using diagnostics might require a decently large sample size. Summary: only use ratios if you really know what you are doing and have good reasons to do it. Otherwise, use linear models.

By the way: we have covered just two very simple examples. Things can of course get a lot more complicated with non-linear relationships, etc.

## 22 Interactions, summary

The general pattern is always the same: the effect of one independent variable (say, A) depends on the setting of the other independent variable (say, B) with which it interacts. In other words, to say something about how a change of variable A affects the outcome, you need to also know the setting or value of variable B.

The three main types are:

**Between factors** There is one level for each combination of the factors, as in the example in section 16.4.

**Betwen a factor and a continuous variable** What we saw in ANCOVA, in section 20: a different slope for each group. (Parallel slopes is not an interaction).

**Between two continuous variables** We mentioned this in section 18.1: slope changes as we move the other variable which gives curved surfaces.

# 23 Diagnostics

Wait!!! Do our models make sense? These are models, so we can, and should, check some of their basic assumptions. We cannot do justice to this **very important topic**.

In general, for linear models (ANOVAs, regressions, etc) we want to check:

- Constant variance (across groups or over the range of the independet variables). Often referred to as "homoscedasticity" (where "heteroscedasticity" is the opposite).
- Linearity (for regression).
- Approximate normality of residuals.
- Possible highly influential points (i.e., do our results depend on one or two points that are driving the model one way or another?).
- Possible outliers.

**Independence** is also a crucial assumption. But often, checking independence is very difficult from the data themselves (or at least from the data we have been using, anyway). For example, lack of independence among data is the reason why the analysis with the AnAge data set are not really correct.

Before looking at the plots, two concepts should be clear:

**Fitted value** The predicted, or fitted value: if we have an equation like $Y = \alpha + \beta_1 X + \beta_2 Z$, then the fitted values are the $Y$ for the observed combinations of $X$ and $Z$ (with the values of $\alpha$ and $\beta$ returned from the model). It is just what the model predicts.

**Residual** Basically, the difference between the observed and the fitted value. There are different types of residuals (residuals, standardized and studentized being the most common).

## 23.1 Diagnostics: an example with a designed experiment with factors

We will first use the fake data from a perfectly balanced experimental design, the data used in section 16.7. The diagnostic plots from these kinds of designed experiments can look slightly different from those for regression, which makes sense if you think about it. I will recreate the data here. However, to make things more interesting, I will create a very large outlier:

```
## Create the data
set.seed(1)
sex <- factor(rep(c("Male", "Female"), c(20, 20)))
drug <- factor(rep(rep(c("A", "B"), c(10, 10)), 2))
y <- rep(c(10, 13, 12, 16), rep(10, 4))
y <- y + rnorm(length(y), sd = 1.5)
y.data <- data.frame(y, sex, drug)
y.data[1, 1] <- 25
## Fit the model
myAdditive <- lm(y ~ sex + drug, data = y.data)
myInteract <- lm(y ~ sex * drug, data = y.data)
## What are they saying?
summary(myAdditive)

##
## Call:
## lm(formula = y ~ sex + drug, data = y.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -4.3955 -1.1085 -0.1430  0.4823 13.9079
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  12.4996     0.7523  16.615  < 2e-16 ***
## sexMale      -1.4075     0.8687  -1.620  0.11368
## drugB         2.9813     0.8687   3.432  0.00149 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.747 on 37 degrees of freedom
## Multiple R-squared:  0.2802,Adjusted R-squared:  0.2413
## F-statistic: 7.201 on 2 and 37 DF,  p-value: 0.002283
```

```
summary(myInteract)
```

```
##
## Call:
## lm(formula = y ~ sex * drug, data = y.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6953 -1.1203 -0.2659  0.8848 13.2077
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   11.799490   0.849216  13.895  4.9e-16 ***
## sexMale       -0.007218   1.200973  -0.006 0.995238
## drugB          4.381605   1.200973   3.648 0.000829 ***
## sexMale:drugB -2.800610   1.698433  -1.649 0.107861
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.685 on 36 degrees of freedom
## Multiple R-squared:  0.3307,Adjusted R-squared:  0.275
## F-statistic:  5.93 on 3 and 36 DF,  p-value: 0.002143
```

```
oldpar <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(myAdditive)
par(oldpar)
```

```
oldpar <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(myInteract)
par(oldpar)
```

First, look at the plots. Try to see what they are about. See how there is one point that stands out in several plots, look at the regularity of the patterns.

The plot on the upper left is used to judge if the functional form of the model makes sense, especially for regression models (not so much for designed experiments with factors, but it is still helpful). This figure also helps spots systematic changes in variance (i.e., violations of homoscedasticity). But for this, the bottom left plot is better which, in this case, does not suggest anything serious, except for the outlier.

The upper right plot (a "q-q plot") is used to assess approximate normality of the residuals: you

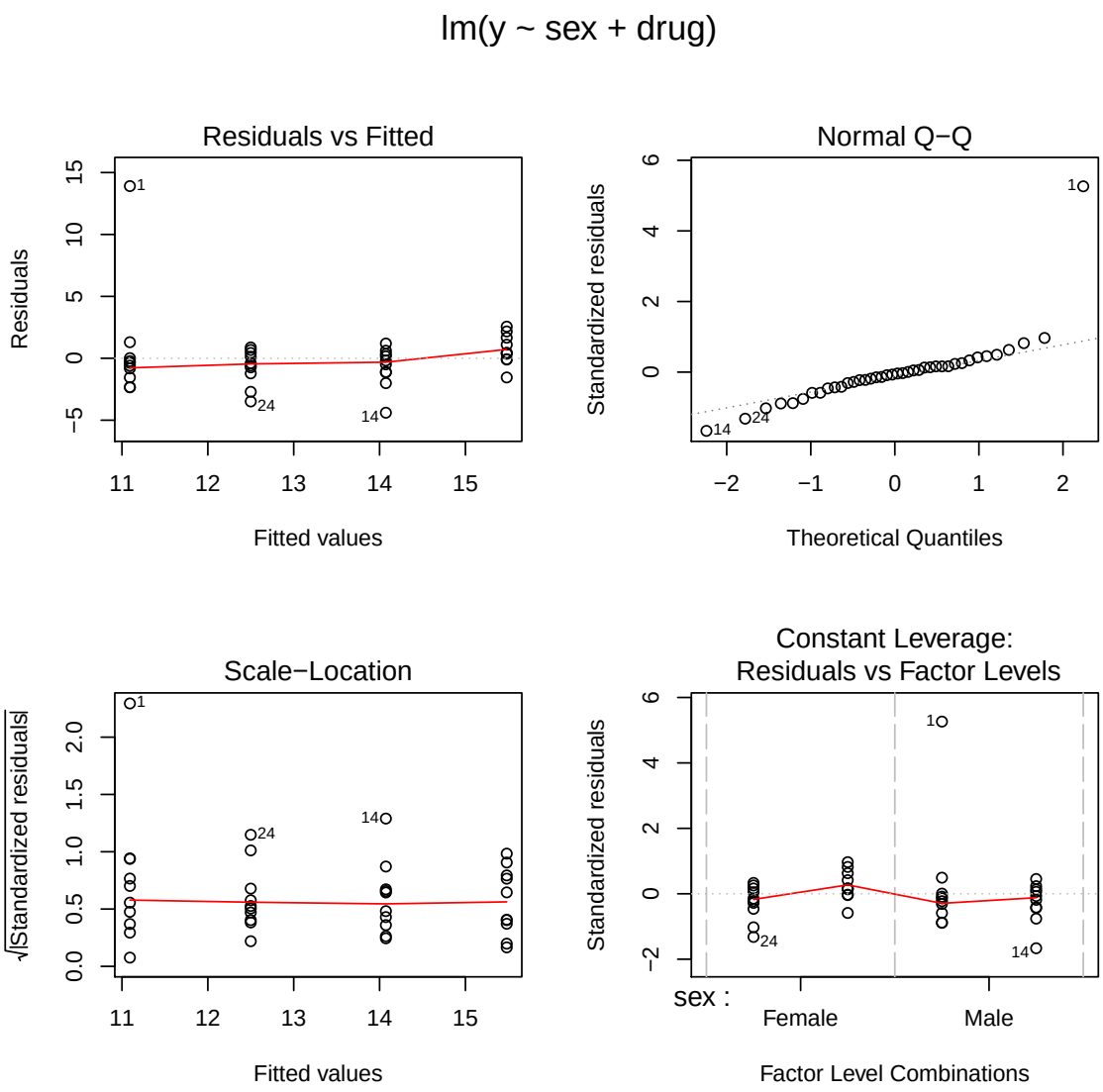



**Figure 4** – Model diagnostics for designed experiment, additive model

want points to more or less lie along the dotted line (with some allowance for deviations in the tails). Here the "q-q plot" is not perfect (even if we discount the outlier), even when data did come from a normal; this is totally normal (remember, we are sampling).

The bottom right differs in regression models and experiments with factors. Here you are shown residuals vs. factor level combinations. (The idea of leverage that makes a lot of sense in regression —see below— is not that useful here). Notice how this plot shows four vertical lines in both models, the additive and the interaction one, but plots on the upper left and bottom left differ in the apparent number of vertical lines. Do you understand why?

## 23.2 Diagnostics: examples with some of the regression models

We will now use the two simple regression models we fitted in section 17. Make the first one (`metab`) active and go to "Models", "Graphs", "Basic diagnostic plots".

```
oldpar <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(metab)
```

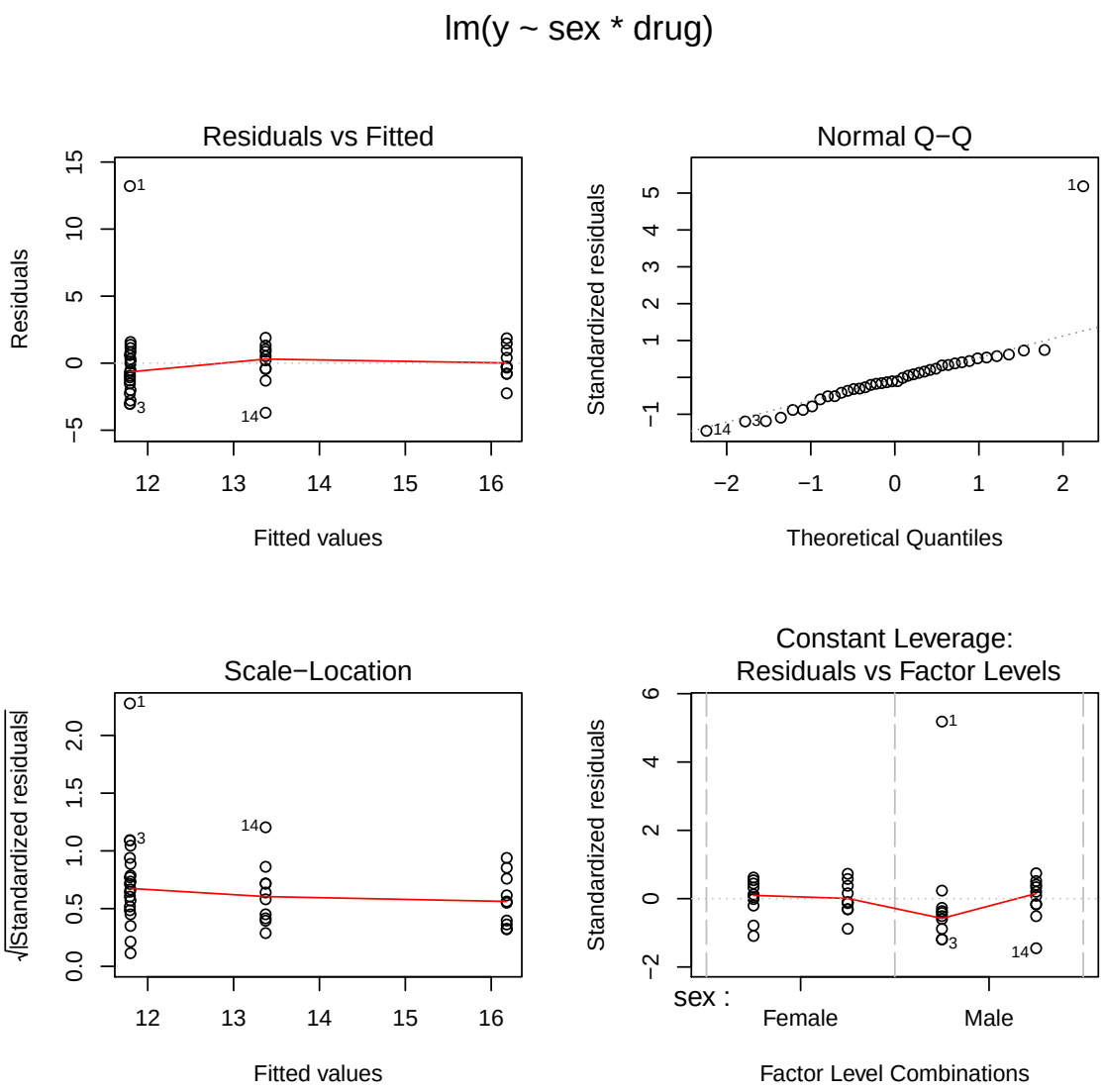



**Figure 5** – Model diagnostics for designed experiment, interaction model

```
par(oldpar)
```

Now you can see why with regression models we can use the plot on the upper left to judge if the functional form of the model makes sense: if the relationship is really linear as modeled, you should see no systematic pattern here, but we see it (in this case, it suggests that out model is predicting too small a metabolic rate at intermediate values, and the opposite at large values, which suggests a curvilinear relationship). Again, this figure also helps spots systematic changes in variance (i.e., violations of homoscedasticity). But for this, the bottom left plot is better and it does suggest that violations of homoscedasticity are present (though, right now, with such strong evidence of non-linearity, this is not a surprise).

What about the "q-q plot": things do not look great here, and this distribution has several very large residuals, is heavy tailed, and also somewhat skewed.

The bottom right can be hard to interpret: it shows two quantities (residuals and leverage) that, together, are part of Cook's distance, that measures the effect of individual points on the fitted coefficients (values of Cook's distance larger than 1 usually indicate a possibly very influential point, but any point with a widely outstanding Cook's distance deserves a closer look). The plot

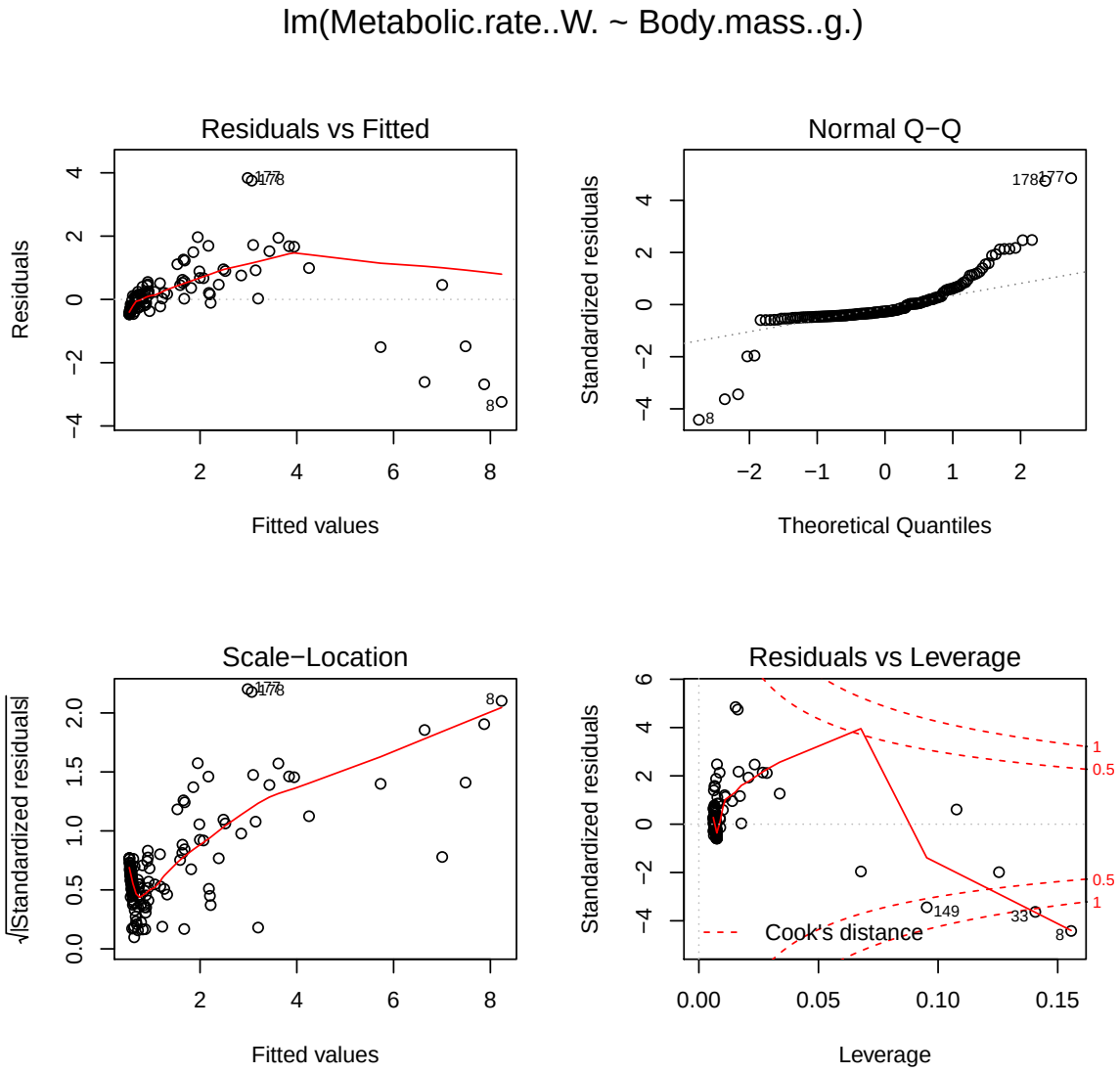



**Figure 6** – Model diagnostics for metabolic rate model without log transformation

called "Influence plots" is very similar to this one. However, I often find it simpler to look at plots Cook's distance (function `cooks.distance`).

Now, repeat the above with the model that has taken logs:

```
oldpar <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(metablog)
par(oldpar)
```

and you will see that all diagnostics look much better.

You should also take a look at the diagnostics for some of the other models we have fitted, specifically `longev_b_r` and `metab_b_r` (or `metab_b_r_2`) (section 20.3). The metabolism one are OK, but the longevity model is not fully satisfactory (small problems in all diagnostic plots, and one potentially influential value). `mcyst2` (section 20) and `mcyst` (section 18), the two models we fitted to the cystic fibrosis data, both look relatively decent, although there could be some concerns about increases in variance with fitted values. However, it is not easy to tell, because of the relatively few points. This, of course, makes sense: if there are few points, we will only be able to detect if a model is a bad one if it really is very bad.

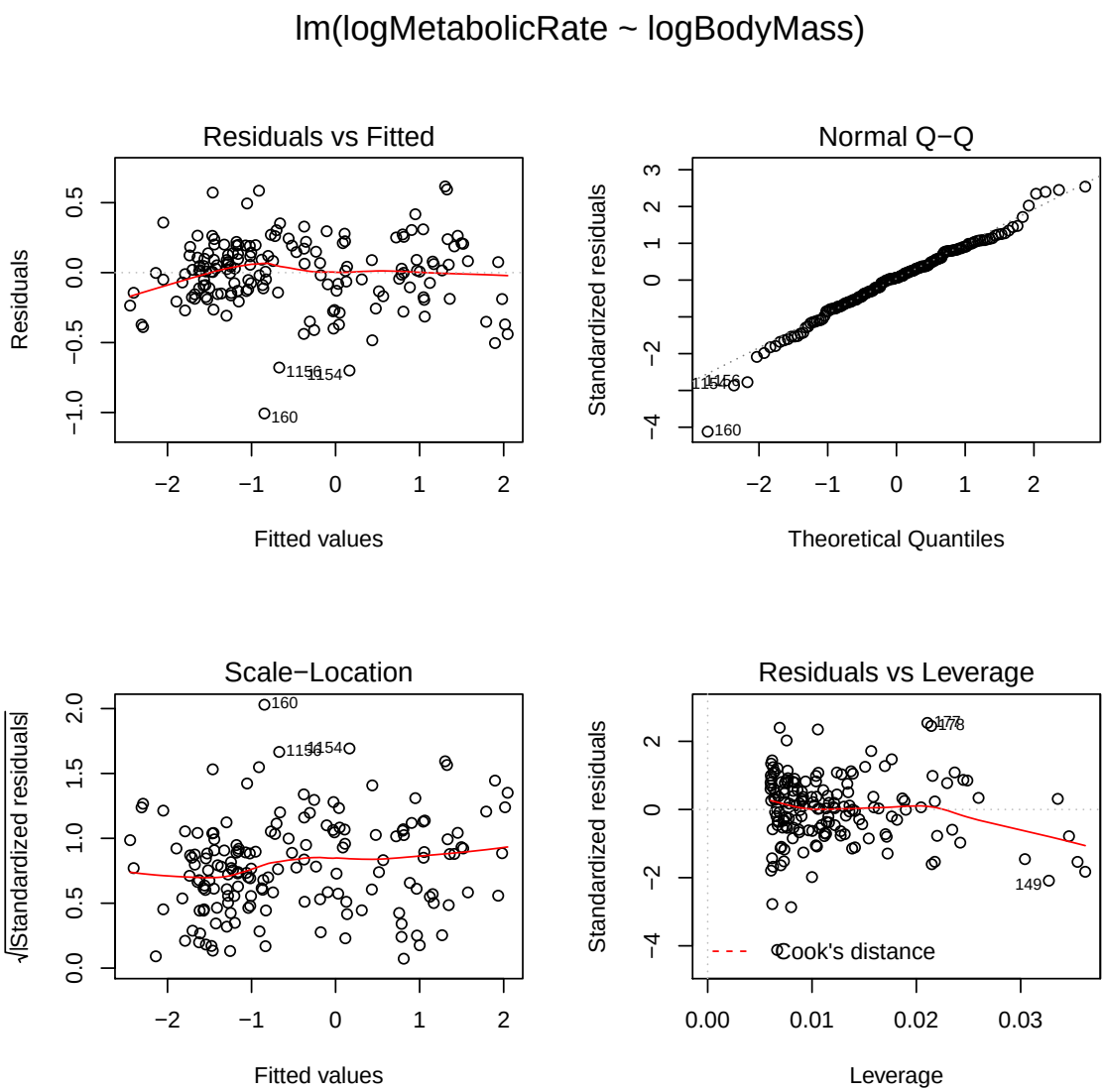



**Figure 7** – Model diagnostics for metabolic rate model after log transformation

`cholestanova`, from section 16.2, looks relatively OK (remember, this is an ANOVA, and there were six combinations of levels, and thus the discrete values you observe in two of the plots). However, the qqplot of residuals is a little bit ugly, but it is hard to tell from relatively so little data[25]. Finally, `AnovaModel.1`, from section 14.3, looks just fine. Please look at all of these yourself to see them.

## 23.3 Diagnostics: a couple more examples with designed experiments

Follow the text and the graphics. We will discuss this in class.

```
## Diagnostics suggest missing interaction
set.seed(1)
sex <- factor(rep(c("Male", "Female"), c(20, 20)))
drug <- factor(rep(rep(c("A", "B"), c(10, 10)), 2))
y <- rep(c(8, 16, 10, 12), rep(10, 4))
```

[25] Again, this is an interesting case, since these data are simulated from a normal distribution

```
y <- y + rnorm(length(y), sd = 1.5)
y.data1 <- data.frame(y, sex, drug)
rm(y, sex, drug)
with(y.data1, tapply(y, list(sex, drug), mean))
```

```
##                A        B
## Female 9.799490 12.18110
## Male   8.198304 16.37327
```

```
## Fit the model
myAdditive2 <- lm(y ~ sex + drug, data = y.data1)
myInteract2 <- lm(y ~ sex * drug, data = y.data1)
summary(myAdditive2)
```

```
##
## Call:
## lm(formula = y ~ sex + drug, data = y.data1)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -3.695 -1.712 -0.232  1.685  3.343
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   8.3512     0.5535  15.088  < 2e-16 ***
## sexMale       1.2955     0.6391   2.027   0.0499 *
## drugB         5.2783     0.6391   8.259 6.42e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.021 on 37 degrees of freedom
## Multiple R-squared:  0.6615,Adjusted R-squared:  0.6432
## F-statistic: 36.16 on 2 and 37 DF,  p-value: 1.979e-09
```

```
summary(myInteract2)
```

```
##
## Call:
## lm(formula = y ~ sex * drug, data = y.data1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6953 -0.6854  0.1639  0.9228  2.1946
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)     9.7995     0.4322  22.676  < 2e-16 ***
## sexMale        -1.6012     0.6112  -2.620 0.012796 *
## drugB           2.3816     0.6112   3.897 0.000407 ***
## sexMale:drugB   5.7934     0.8643   6.703 8.08e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.367 on 36 degrees of freedom
```

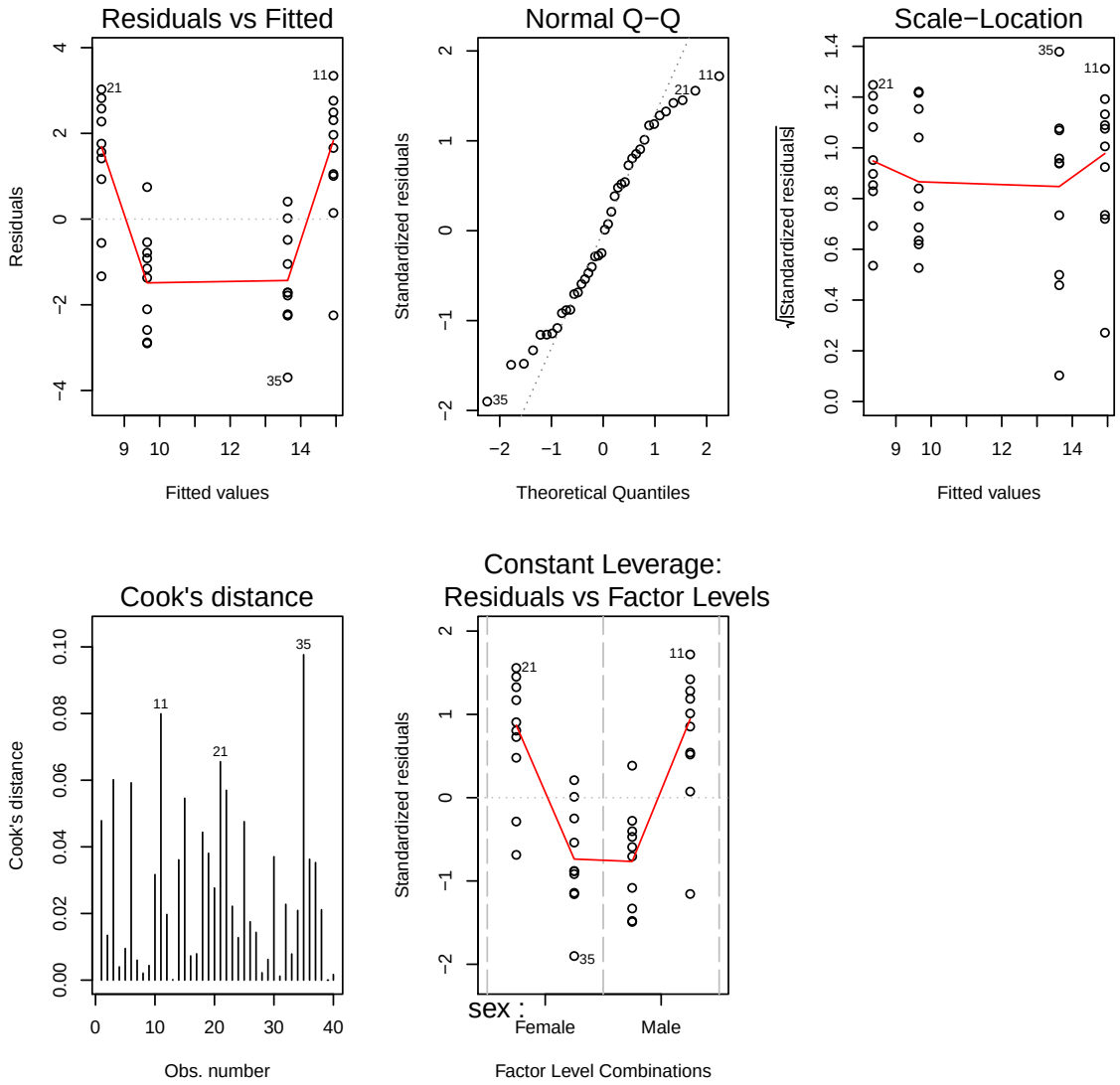



**Figure 8** – Additive model, myAdditive2, when there is interaction

```
## Multiple R-squared:  0.8494,Adjusted R-squared:  0.8369
## F-statistic:  67.7 on 3 and 36 DF,  p-value: 7.158e-15
```

```
par(mfrow = c(2, 3))
plot(myAdditive2, which = c(1:5)) ## look at first plot
```

```
par(mfrow = c(2, 3))
plot(myInteract2, which = c(1:5))
```

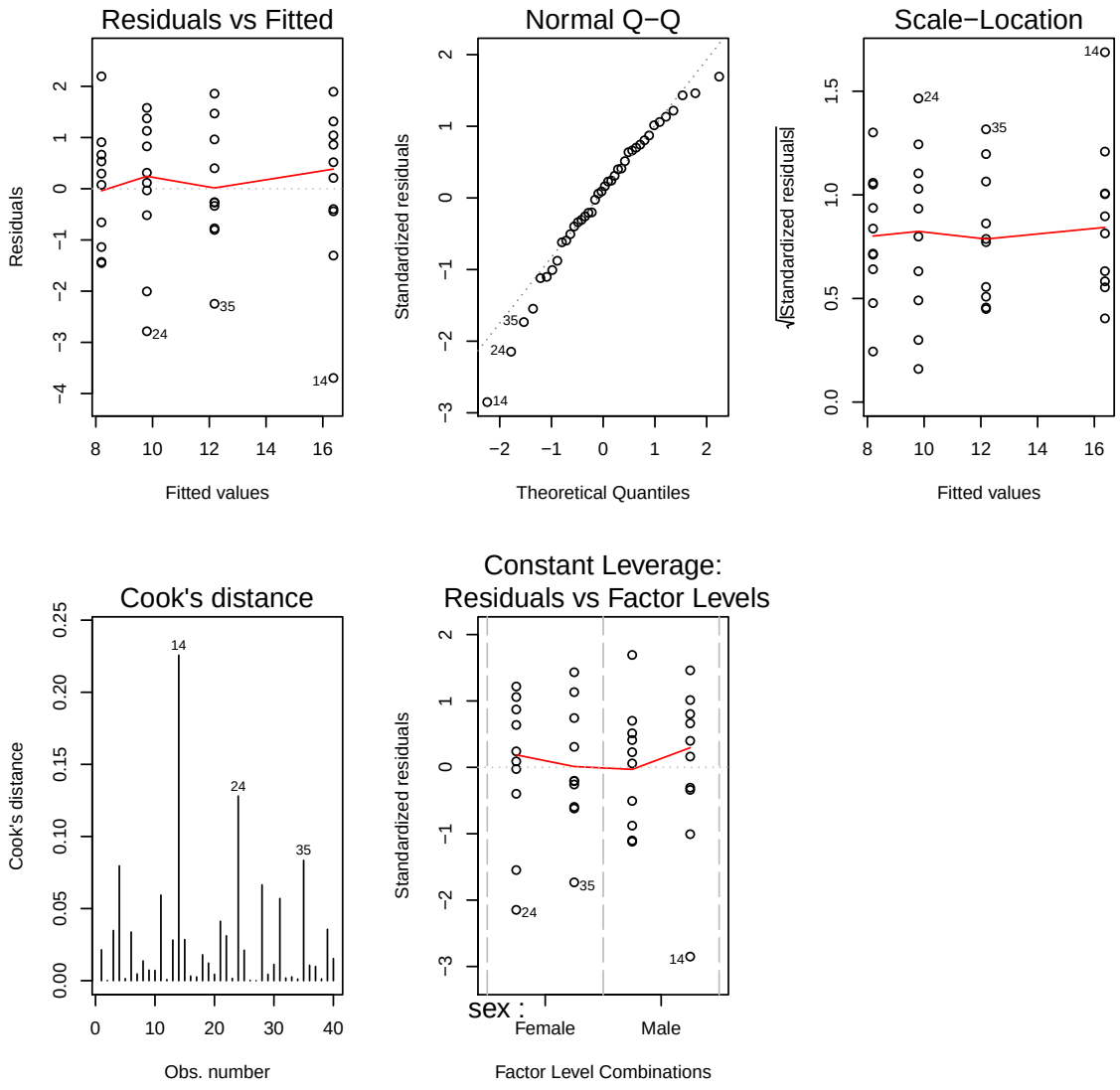



**Figure 9** – Interaction model, myInteract2, when there is interaction

```
############
## Large cook's in anova
set.seed(1)
sex <- factor(rep(c("Male", "Female"), c(20, 20)))
drug <- factor(rep(rep(c("A", "B"), c(10, 10)), 2))
y <- rep(c(8, 12, 11, 15), rep(10, 4))
y <- y + rnorm(length(y), sd = 1.5)
y.data2 <- data.frame(y, sex, drug)
rm(y, sex, drug)
## create a large outlier

y.data2[1, 1] <- 30
with(y.data2, tapply(y, list(sex, drug), mean))
```

```
##                A        B
## Female 10.79949 15.18110
## Male   10.49227 12.37327
```

```
## ## Fit the model
myAdditive2b <- lm(y ~ sex + drug, data = y.data2)
## myInteract <- lm(y ~ sex * drug, data = y.data2)
## summary(myAdditive)
## summary(myInteract)
```

```
## ## diagnostics, all of them except 6th
## we actually have a large Cook's distance
par(mfrow = c(2, 3))
plot(myAdditive2b, which = 1:5)
```

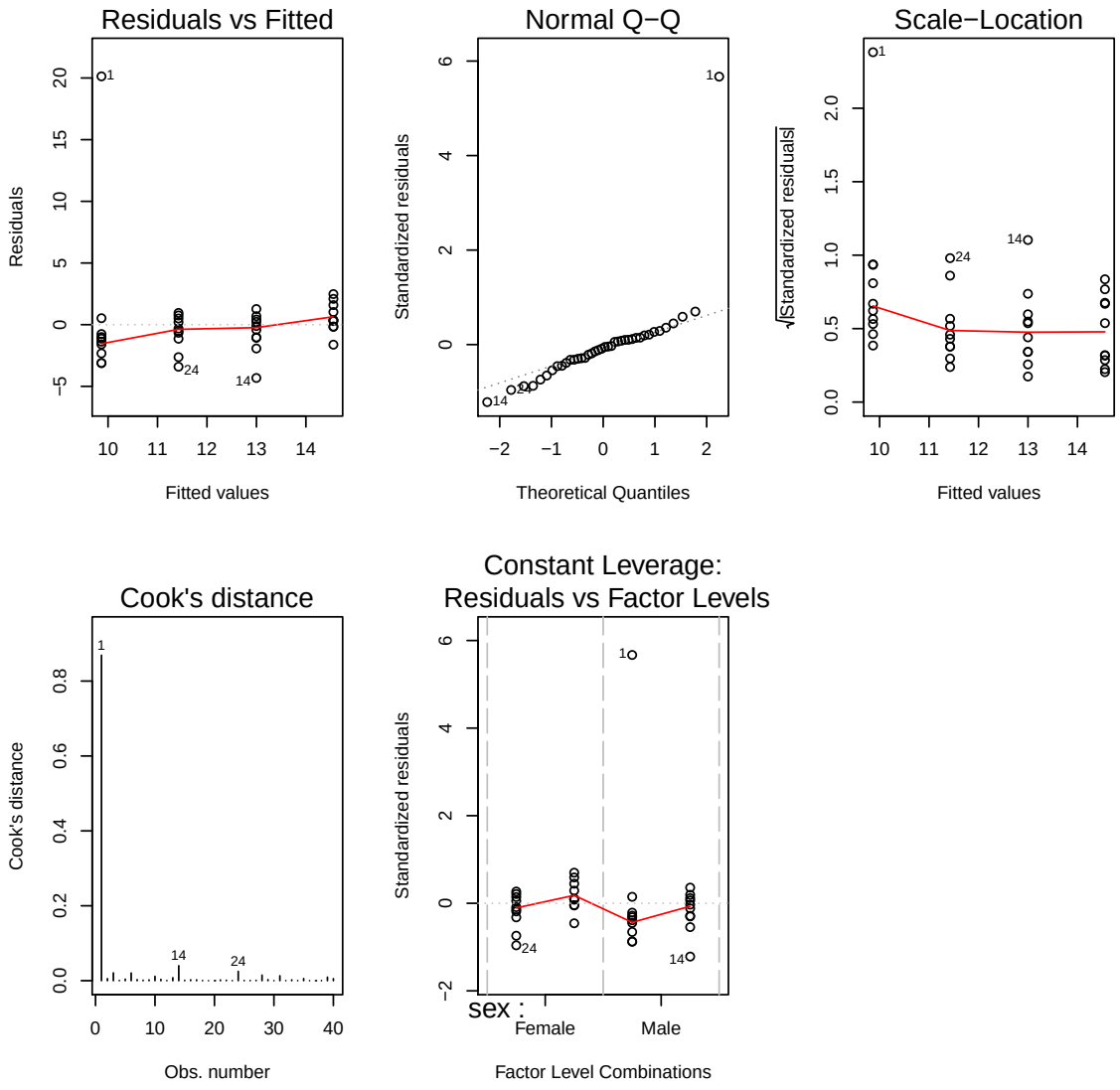



**Figure 10** – myAdditive2b: large Cook with balanced data

```
## model with and without first obs
summary(lm(y ~ sex + drug, data = y.data2))

##
## Call:
## lm(formula = y ~ sex + drug, data = y.data2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.3205 -1.1996 -0.2456  0.5103 20.1329
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   11.425      1.011  11.300 1.48e-13 ***
## sexMale       -1.558      1.167  -1.334   0.1903
## drugB          3.131      1.167   2.682   0.0109 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.692 on 37 degrees of freedom
## Multiple R-squared:  0.1952,Adjusted R-squared:  0.1517
## F-statistic: 4.487 on 2 and 37 DF,  p-value: 0.018

summary(lm(y ~ sex + drug, data = y.data2[-1, ]))

##
## Call:
## lm(formula = y ~ sex + drug, data = y.data2[-1, ])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.7763 -0.6446  0.1305  0.9171  2.1582
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  10.8805     0.3727  29.197  < 2e-16 ***
## sexMale      -2.6458     0.4341  -6.094 5.20e-07 ***
## drugB         4.2196     0.4341   9.720 1.32e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.355 on 36 degrees of freedom
## Multiple R-squared:  0.7813,Adjusted R-squared:  0.7691
## F-statistic: 64.29 on 2 and 36 DF,  p-value: 1.314e-12
```

```
#### Diagnostics if we remove the offending value
par(mfrow = c(2, 3))
plot(lm(y ~ sex + drug, data = y.data2[-1, ]), which = 1:5)
```

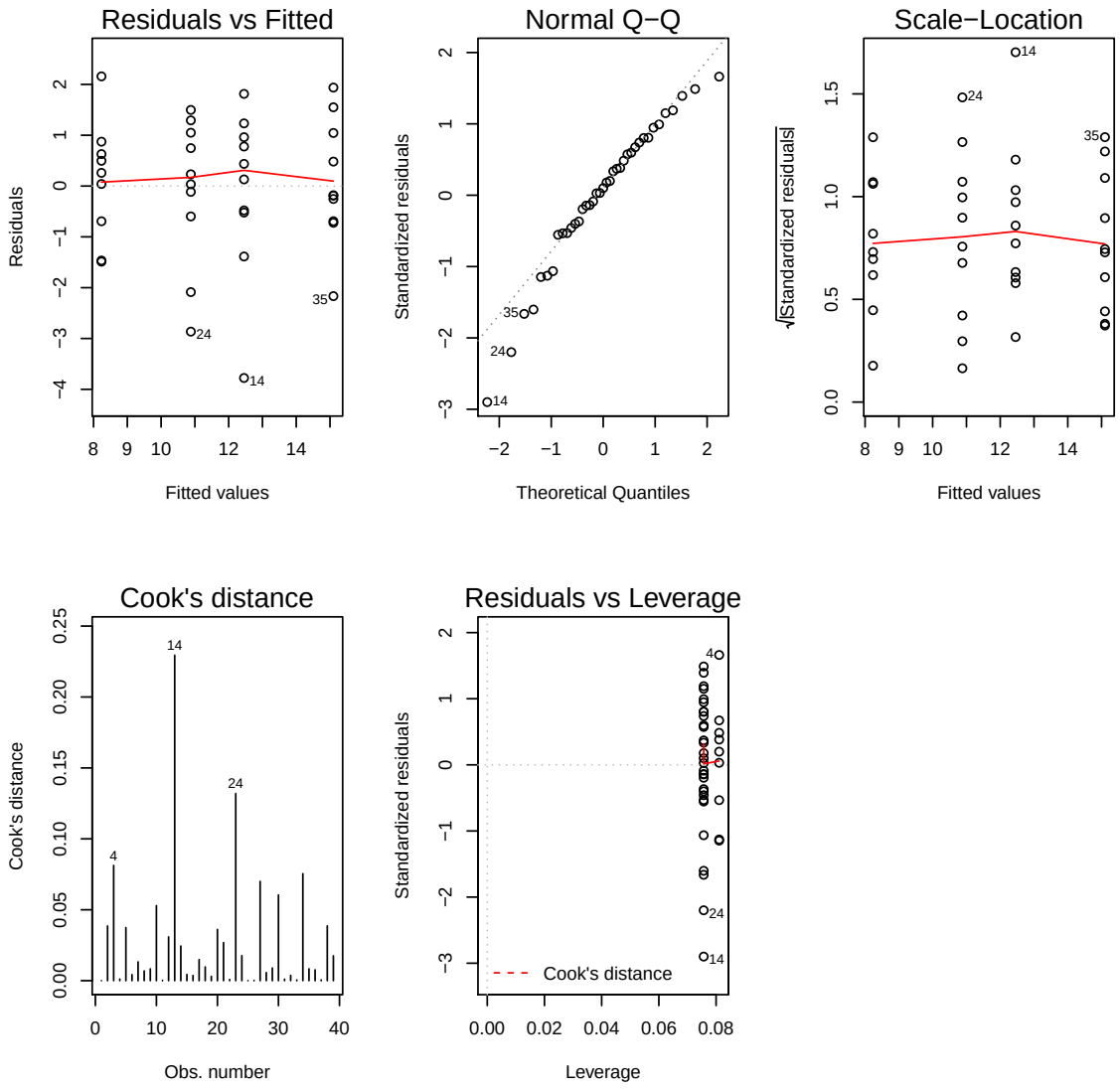



**Figure 11** – myAdditive2b removing the large offending value

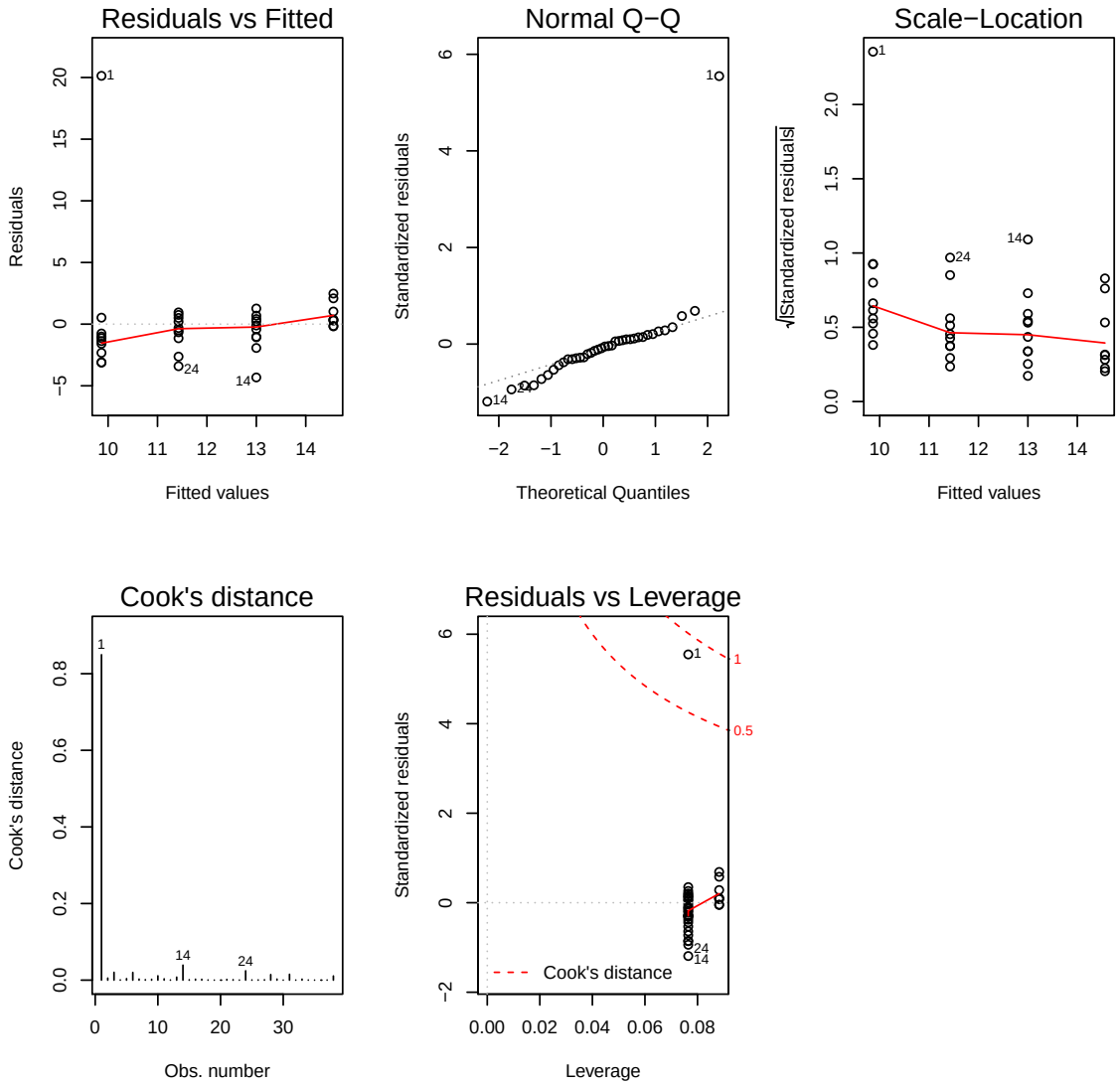



**Figure 12** – myAdditive3: missing two observations

```
## The model with two observations missing
## create unbalance
y.data3 <- y.data2[-c(35, 40), ]
with(y.data3, tapply(y, list(sex, drug), mean))
```

```
##                A        B
## Female 10.79949 15.34147
## Male   10.49227 12.37327
```

```
myAdditive3 <- lm(y ~ sex + drug, data = y.data3)
```

```
par(mfrow = c(2, 3))
plot(myAdditive3, which = 1:5) ## see Cook's
```

```
## some details, from plot.lm
(hii2b <- lm.influence(myAdditive2b, do.coef = FALSE)$hat) ## constant
(hii3 <- lm.influence(myAdditive3, do.coef = FALSE)$hat) ##nope
```

```
diff(range(hii2b))
```

```
## [1] 5.551115e-17
```

```
diff(range(hii3))
```

```
## [1] 0.01176471
```

### 23.4 Diagnostics: further issues

For you to read and play around further:

- An extended qqplot is available from package *car*, `qqPlot` (and in R Commander under "Residual quantile comparison plots").
- "Component+Residual" (or partial residual[26]) plots allow us to examine, in models with multiple regressors, deviations from linearity and could suggest the appropriate transformation. "CERES plots" are a variation of "Component + Residual" plots that work well even if relationships are strongly nonlinear. Both also available from *car*.
- Various diagnostics related to dfbetas allow us to identify influential observations in specific terms of the model.
- Variance inflation factors help us detect possible problems caused by collinearity (correlations between independent variables).
- Added variable plots are particularly useful in multiple regression problems with multiple independent variables; they can help to identify influential points (which are easily masked with multiple variables) and can also help to try to find a good functional relationship (but Component+Residual are more useful here). Again, available from *car*.
- A variety of numerical tests and diagnostics are also available (e.g., tests for nonlinearity or for homoscedasticity).
- Package *car* and the accompanying book "An R companion to applied regression" contain excellent and detailed comments about those and other diagnostics, and several functions. Take also a look, for a summary, at the very nice sections 8.3 to 8.5 in Kabakoff's "R in action."

## 24 Variable and model selection

You say you want to select variables? Select them for what? Prediction? Interpretation? Variable selection is a touchy and delicate subject. First, procedures based purely in statistical criteria might select "statistically important" variables (under some suitable definition of "important"), but those need not be the most relevant from a biological point of view, or causally, etc

And with regards to the statistical procedures, we will summarize it as follows: please, please, please, distrust automated variable selection procedures that rely on p-values or F-statistics of individual variables (in all their variants, such as stepwise, etc). Careful model comparison, for instance using something like `anova(model1, model2)`, as we have seen (e.g., section 20.2 or 20.3), might be a good idea. Notice, again, that `anova(model1, model2)` is all about comparing models.

[26] These plots, for each variable, shows a plot of the independent variable, say $x$, on the horizontal axis and, on the ordinates, the "partial residuals" $= residual + \hat{\beta}x$.

If you really need automated or semiautomated procedures, then reasonable strategies use model-comparison criteria such as AIC (e.g., with the `step` function) and even better is then bootstrapping the whole process to get estimates of error, predictive ability, etc. Even better, of course, is subject-matter guidance on how to proceed and what are and are not candidates for deletion and in what order. The book by Frank Harrell, "Regression modeling strategies" contains great discussions of these topics. The actual data set used here is discussed, also in the context of variable selection, by P. Dalgaard in chapter 11 of "Introductory statistics with R".

To give some examples (look at them carefully, and compare with what we did by hand, and with whether or not these are the models you would use):

```
step(mcyst2, direction = "both")
```

```
## Start:  AIC=168.89
## pemax ~ age * sex
##
##           Df Sum of Sq   RSS    AIC
## - age:sex  1       189 15779 167.19
## <none>                 15590 168.89
##
## Step:  AIC=167.19
## pemax ~ age + sex
##
##           Df Sum of Sq   RSS    AIC
## - sex      1     955.4 16734 166.66
## <none>                 15779 167.19
## + age:sex  1     189.0 15590 168.89
## - age      1    8819.5 24598 176.29
##
## Step:  AIC=166.66
## pemax ~ age
##
##        Df Sum of Sq   RSS    AIC
## <none>              16734 166.66
## + sex   1     955.4 15779 167.19
## - age   1   10098.5 26833 176.46
##
## Call:
## lm(formula = pemax ~ age, data = cystfibr2)
##
## Coefficients:
## (Intercept)          age
##      50.408        4.055
```

```
step(mcyst, direction = "both")
```

```
## Start:  AIC=169.19
## pemax ~ age + height + weight
##
##          Df Sum of Sq   RSS    AIC
## - height  1      6.75 15789 167.21
## - age     1    186.86 15969 167.49
## - weight  1    769.60 16552 168.38
## <none>                 15782 169.19
##
```

```
## Step:  AIC=167.2
## pemax ~ age + weight
##
##          Df Sum of Sq   RSS    AIC
## - age     1    216.51 16006 165.54
## - weight  1    945.19 16734 166.66
## <none>                 15789 167.21
## + height  1      6.75 15782 169.19
##
## Step:  AIC=165.55
## pemax ~ weight
##
##          Df Sum of Sq   RSS    AIC
## <none>                 16006 165.54
## + age     1     216.5 15789 167.21
## + height  1      36.4 15969 167.49
## - weight  1   10827.2 26833 176.46
##
## Call:
## lm(formula = pemax ~ weight, data = cystfibr2)
##
## Coefficients:
## (Intercept)       weight
##      63.546        1.187
```

```
## In metab, we drop the interaction
step(metab_b_r, direction = "both")
```

```
## Start:  AIC=-463.77
## logMetabolicRate ~ logBodyMass * Class
##
##                      Df Sum of Sq    RSS     AIC
## - logBodyMass:Class  1  0.075167 12.646 -464.71
## <none>                           12.571 -463.77
##
## Step:  AIC=-464.71
## logMetabolicRate ~ logBodyMass + Class
##
##                      Df Sum of Sq     RSS     AIC
## <none>                             12.646 -464.71
## + logBodyMass:Class  1     0.075  12.571 -463.77
## - Class              1   109.234 121.880  -63.42
## - logBodyMass        1   226.058 238.704   56.23
##
## Call:
## lm(formula = logMetabolicRate ~ logBodyMass + Class, data = anage_a_r)
##
## Coefficients:
##   (Intercept)    logBodyMass  ClassReptilia
##       -3.1460         0.6471        -3.1885
```

```
## But nothing can be dropped here
step(longev_b_r, direction = "both")
```

```
## Start:  AIC=-308.85
## logLongevity ~ logBodyMass * Class
##
##                     Df Sum of Sq    RSS     AIC
## <none>                           27.258 -308.85
## - logBodyMass:Class  1    1.6414 28.900 -300.79
##
## Call:
## lm(formula = logLongevity ~ logBodyMass * Class, data = anage_a_r)
##
## Coefficients:
##               (Intercept)                logBodyMass              ClassReptilia
##                    1.7888                     0.2182                    -0.9858
## logBodyMass:ClassReptilia
##                    0.2561
```

## 25 Experimental design matters

All of the data we have seen so far have been relatively straightforward. Things aren't always this way. Suppose a situation such as this:

- 20 mice.
- 10 assigned to drug A, 10 assigned to drug B.
- Each mouse in one leg gets a corticoid ointment, on another leg gets a placebo ointment.
- What is the experimental unit?

There are, in fact, two experimental units: mouse and leg within mouse. To compare drugs we use mice. To compare ointment: we should use leg within mouse. Interactions? Can be studied, yes. How do we analyze this? This example, nicely balanced, is a classical example of a split-plot design (a type of ANOVA with multiple strata). But designs like this, and others more general, or like this but unbalanced, or like this but with additional covariates, etc, are nowadays analyzed using mixed-effects models.

This also relates to questions we asked in section on "Non-independent data" 12: What if we had repeated measures on the same subjects over time? Or if we had some data that came from brothers, cousins, etc?

And how would you go about designing an experiment from scratch? What should you randomize over and what should you block over? Should you use a factorial design? Matched pairs? Should each one of two technicians each take care of half of the samples, randomly assigned to each, or should one technician deal with all male sample and the other with all the female samples? Should we start mice in each one of the four different diets in different days of the week, or should we have similar sized groups of each diet starting every day of the week? Do you give treatment A to all even numbered samples and treatment B to all odd numbered ones, or do you randomize order? Etc, etc. And, of course, how should we allocate sample sizes to the different **levels of variation**?

Understanding what is the experimental unit is **absolutely crucial**. And sometimes things are complicated: talk to (collaborate with) a statistician as soon as you can. Some high-profile mistakes in the literature are derived from misunderstanding experimental design (a somewhat amusing half-page comment about this by G. Churchill in *Science*, 2013, v. 343, p. 370). In fact,

some mistakes made during the experimental design phase just cannot be corrected later[27].

## 26 Additional reading and what next

Many books have been devoted to linear models, ANOVA, et al. And in R (not necessarily through R Commander) there is a wide variety of procedures implemented. To begin with, look at the great book by Fox and Weisberg "An R companion to applied regression". Take also a look at chapters 6 and 7 of Dalgaard's "Introductory statistics with R" and chapters 8 and 9 of Kabakoff's "R in action". This should get you going. From here, you will probably want to look at generalized linear models, survival analysis, mixed effects models, nonlinear models, and generalized additive models to name some extensions of linear models.

## A What if we did not recode training?

Suppose we had not recoded training but we had fitted a linear model. This would have happened:

```
lmMITnofactor <- lm(activ ~ training, data = dmit)
summary(lmMITnofactor)

##
## Call:
## lm(formula = activ ~ training, data = dmit)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.61093 -0.72914 -0.06266  0.64855  1.86234
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.8876     0.3810   2.330   0.0245 *
## training      0.9234     0.1584   5.829 6.02e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8802 on 44 degrees of freedom
## Multiple R-squared:  0.4357,Adjusted R-squared:  0.4229
## F-statistic: 33.97 on 1 and 44 DF,  p-value: 6.024e-07

Anova(lmMITnofactor)

## Anova Table (Type II tests)
##
## Response: activ
##           Sum Sq Df F value    Pr(>F)
## training  26.320  1  33.973 6.024e-07 ***
## Residuals 34.088 44
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

[27] R. Fisher, one of the fathers of modern statistics, as well as modern quantitative and evolutionary genetics, is often quoted in this context because he said "To consult the statistician after an experiment is finished is often merely to ask him to conduct a post mortem examination. He can perhaps say what the experiment died of."

See how the degrees of freedom for training make no sense.

# B Session info and packages used

This is the information about the version of R and packages used when producing this document:

```
sessionInfo()
```

```
## R Under development (unstable) (2016-12-19 r71815)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Debian GNU/Linux stretch/sid
##
## locale:
##  [1] LC_CTYPE=en_GB.utf8       LC_NUMERIC=C
##  [3] LC_TIME=en_GB.utf8        LC_COLLATE=en_GB.utf8
##  [5] LC_MONETARY=en_GB.utf8    LC_MESSAGES=en_GB.utf8
##  [7] LC_PAPER=en_GB.utf8       LC_NAME=C
##  [9] LC_ADDRESS=C              LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_GB.utf8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] grid      tools     stats     graphics  grDevices
## [6] utils     datasets  base
##
## other attached packages:
##  [1] HH_3.1-32           gridExtra_2.2.1
##  [3] latticeExtra_0.6-28 RColorBrewer_1.1-2
##  [5] lattice_0.20-34     effects_3.1-2
##  [7] multcomp_1.4-6      TH.data_1.0-7
##  [9] MASS_7.3-45         survival_2.40-1
## [11] mvtnorm_1.0-5       tidyr_0.6.0
## [13] RcmdrMisc_1.0-5     sandwich_2.3-4
## [15] car_2.1-4           BiocStyle_2.3.26
## [17] patchSynctex_0.1-4  stringr_1.1.0
## [19] knitr_1.15.1
##
## loaded via a namespace (and not attached):
##  [1] vcd_1.4-3          base64_2.0
##  [3] Rcpp_0.12.8        class_7.3-14
##  [5] zoo_1.7-14         gmp_0.5-12
##  [7] lmtest_0.9-34      assertthat_0.1
##  [9] rprojroot_1.1      digest_0.6.10
## [11] mime_0.5           R6_2.2.0
## [13] plyr_1.8.4         backports_1.0.4
## [15] acepack_1.4.1      MatrixModels_0.4-1
## [17] evaluate_0.10      e1071_1.6-7
## [19] ggplot2_2.2.0      highr_0.6
## [21] lazyeval_0.2.0     readxl_0.1.1
## [23] minqa_1.2.4        data.table_1.10.0
## [25] SparseM_1.74       nloptr_1.0.4
## [27] rpart_4.1-10       Matrix_1.2-7.1
## [29] rmarkdown_1.2      splines_3.4.0
```

```
## [31] lme4_1.1-12         foreign_0.8-67
## [33] munsell_0.4.3       shiny_0.14.2
## [35] httpuv_1.3.3        compiler_3.4.0
## [37] mgcv_1.8-16         htmltools_0.3.5
## [39] nnet_7.3-12         openssl_0.9.5
## [41] tibble_1.2          htmlTable_1.7
## [43] Hmisc_4.0-1         codetools_0.2-15
## [45] leaps_2.9           xtable_1.8-2
## [47] nlme_3.1-128        gtable_0.2.0
## [49] magrittr_1.5        scales_0.4.1
## [51] stringi_1.1.2       Rmpfr_0.6-1
## [53] reshape2_1.4.2      Formula_1.2-1
## [55] abind_1.4-5         parallel_3.4.0
## [57] pbkrtest_0.4-6      yaml_2.1.14
## [59] colorspace_1.3-2    cluster_2.0.5
## [61] methods_3.4.0       quantreg_5.29
```