R workshop #3: Mixed-effects models

Nicola Romanò

Introduction

As we have seen in the lectures, both repeated measures and nested designs pose some challenges to data analysis. In particular, these types of design are problematic as they contain correlated observations.

For example, in the case of a repeated measure design, where we measure a certain parameter in the same subject at different times, each of the measurements is likely to be dependent (thus correlated to some degree) on the previous value.

One of the best ways of dealing with these data is to use mixed effects models, an extension of the linear model that allows to account for random effects into the model.

For the context of this workshop we will need to use the `nlme` R package[1]. As usual, this can be installed using the following command:

```
install.packages("nlme")
```

Installation needs to be performed only once. The package can then be used after loading using:

```
library(nlme)
```

Note: mixed-effects models are quite a complex tool to use. The aim of this workshop is to introduce you to these models, and give you an idea of how to use them for some basic analysis. This is far from being a comprehensive tool[2], and analysis of more complex designs may be not so trivial!

Learning objectives

After completing this workshop you will be able to:

- Creat a mixed effect model to explore simple repeated measure or nested designs.
- Interpret the output of a mixed effect model

Repeated Measure Design

Let's consider this simple repeated measure design.

We are interested in evaluating the effect of the dopaminergic agonist bromocriptine on the growth of prolactinomas (pituitary

[1] The other commonly used package is `lme4`, with the `lmer` function. The syntax is slightly different but similar reasoning apply

[2] If you are really interested to expand your knowledge in this area, a very good book is "Mixed Effects Models in S and S-PLUS" by Pinheiro and Bates (this uses S, which is the language R derives from; the S code can be run in R with no major issue)

adenomas secreting prolactin). We take 45 mice and randomly assign them to one of three groups (15/group).

The control group receives a sham surgery, while the other two groups receive a subcutaneous implant containing either 1 or 10 mg bromocriptine. We then follow the adenoma growth over time by measuring plasma levels of prolactin (PRL).

We have two fixed effects in this design: time and treatment, and a random effect, the mouse. Since each mouse is measured several times, measurements coming from the same animal will not be independent, thus the need for a mixed effect model.

For this example, we start by loading the file `bromocriptine-workshop3.csv`

```
bromocriptine <- read.csv("bromocriptine-workshop3.csv")
```

```
summary(bromocriptine)
```

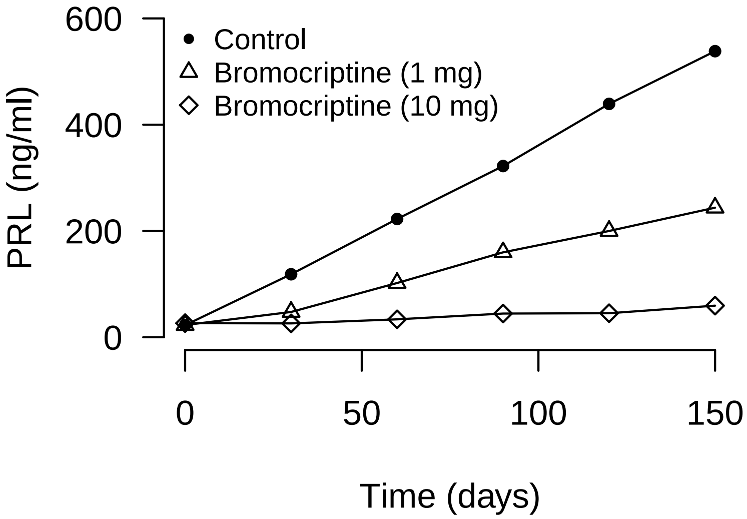
```
##       Time          Mouse         PRL              Group   
##  Min.   :  0   Min.   : 1   Min.   :  0.4252   Bromo1 :90  
##  1st Qu.: 30   1st Qu.:12   1st Qu.: 31.4421   Bromo10:90  
##  Median : 75   Median :23   Median : 78.0624   CTRL   :90  
##  Mean   : 75   Mean   :23   Mean   :148.6180               
##  3rd Qu.:120   3rd Qu.:34   3rd Qu.:222.3295               
##  Max.   :150   Max.   :45   Max.   :775.6684               
```

As usual, explore the data, try to plot it and see possible relationships between variables[3].

I will also rearrange the order of the levels in the Group factor so that the control group is used as the reference level.

```
bromocriptine$Group <- factor(bromocriptine$Group,
    levels = c("CTRL", "Bromo1", "Bromo10"))
```

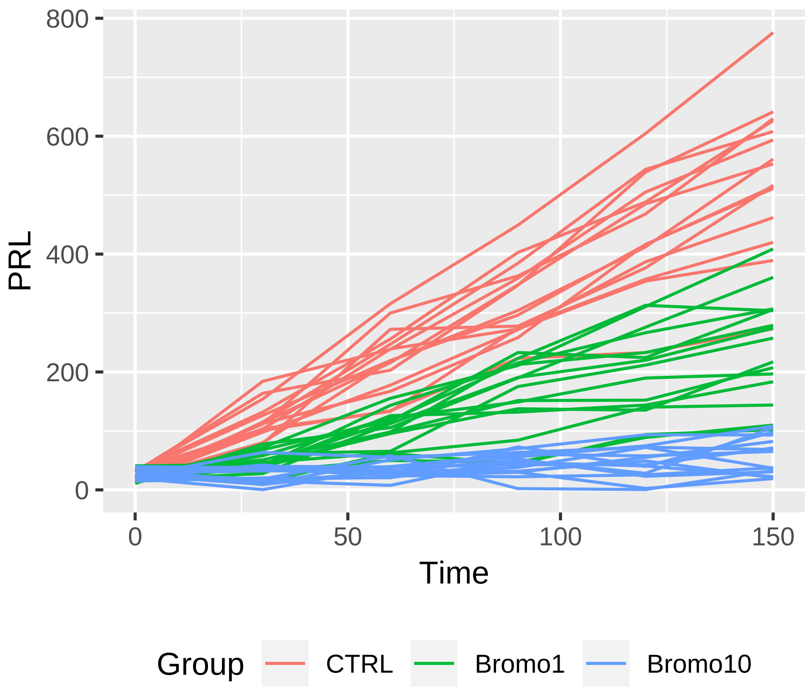
Below I have plotted the mean PRL values over time in the three groups[4]



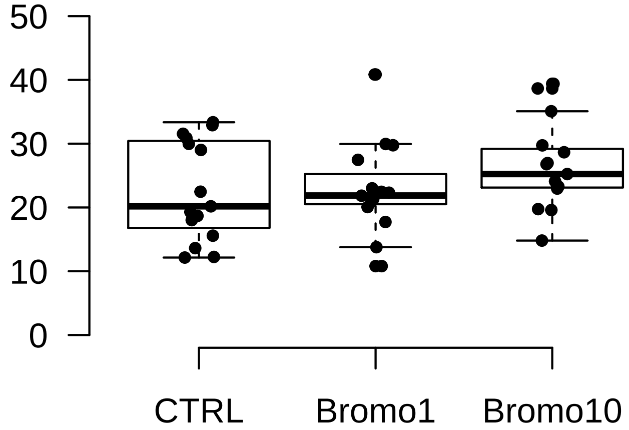
It is clear that there is a dose-dependent effect of the drug, and that, overall, the change of PRL levels over time can be studied using a linear model. If we look at the responses of single mice we can see that they start more or less at the same level, but they then rise in different ways (with different slopes) depending on the treatment, but also within the treatment.

[3] Refer to previous workshops if you do not remember how

[4] As a challenge, try to reproduce this graph, you may post your code on the discussion board!



We can see individual mice differences are not too pronounced at time 0.



Let's now proceed and build our model [5]. We model the fixed effects Group and Time, as well as their interaction, and we model Mouse as the random factor. We create a random slope model, since what seems to be variable amongst subjects is the slope. We specify the random effect as `Time - 1 | Mouse`, meaning that we want to use mouse as a random effect, and we want to have random slopes over Time but not random intercepts (hence the -1).

```
model <- lme(PRL ~ Group * Time, data = bromocriptine, random = ~Time - 1 | Mouse)
summary(model)
```

```
## Linear mixed-effects model fit by REML
##  Data: bromocriptine
##        AIC     BIC    logLik
##   2564.473 2593.08 -1274.236
##
## Random effects:
##  Formula: ~Time - 1 | Mouse
##            Time Residual
## StdDev: 0.54899 20.62831
##
## Fixed effects: PRL ~ Group * Time
##                        Value Std.Error  DF    t-value p-value
## (Intercept)        17.239483  3.854825 222   4.472183  0.0000
## GroupBromo1        -3.546567  5.451545  42  -0.650562  0.5189
## GroupBromo10        5.390355  5.451545  42   0.988776  0.3284
## Time                3.466547  0.147966 222  23.428049  0.0000
## GroupBromo1:Time   -1.924222  0.209255 222  -9.195581  0.0000
## GroupBromo10:Time  -3.244864  0.209255 222 -15.506739  0.0000
##  Correlation:
##                    (Intr) GrpBr1 GrpB10 Time   GrB1:T
## GroupBromo1        -0.707
## GroupBromo10       -0.707  0.500
## Time               -0.237  0.167  0.167
## GroupBromo1:Time    0.167 -0.237 -0.118 -0.707
## GroupBromo10:Time   0.167 -0.118 -0.237 -0.707  0.500
##
## Standardized Within-Group Residuals:
##         Min          Q1         Med          Q3         Max
## -2.78846661 -0.59600115  0.02844046  0.52908281  3.02785483
##
## Number of Observations: 270
## Number of Groups: 45
```

This is a fairly complex model, the important parts that we want to look at in the summary are:

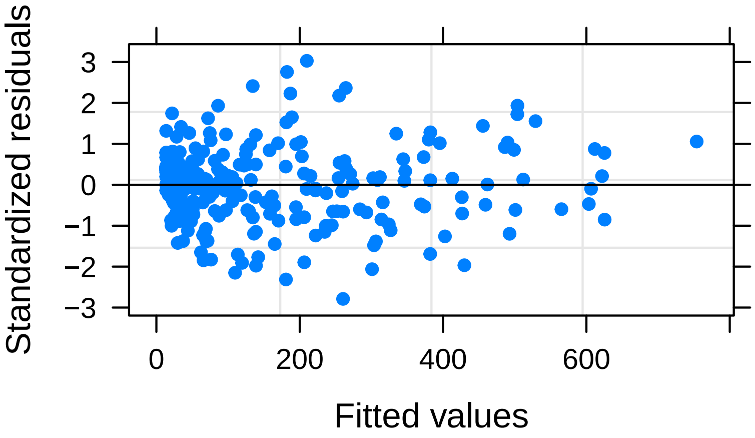
- AIC, BIC and logLik: these are goodness-of-fit measure. For both AIC and BIC the smaller the better, for the log-likelyhood, the higher the better.

[5] Remember to load the nlme package first!

- The random effect part gives us the standard deviations for the random effects and the residuals (remember from the lectures, these derive from two independent normal distribution).
- The fixed effect part gives us the estimates for the model coefficients. In this case, the intercept 17.23 is the PRL level for a mouse in the control group (because this is the reference level of our Group variable), at time 0. The other coefficients are interpreted as we have seen in workshop #2. Note that there is, as expected, a strong interaction between time and treatment, i.e. PRL varies differently over time for different treatments.
- A correlation table. For most, if not all, of the situations you will be dealing with you can safely ignore this.
- A summary of the distribution of residuals
- The number of observations and groups. You can use this to check that you have correctly specified your experiment structure in the model. We have a total of 270 observations (you can check this by running `ncol(bromocriptine)`) and 45 groups (i.e. experimental units), corresponding to the 45 mice. This tells us that R has understood that those 270 observations are not independent, but come from 45 mice, therefore multiple observations are associated with the same mouse.

We can check the distribution of residuals over the fitted values as usual (although the output has a slight different format compared to `lm`)

```
plot(model, pch = 20)
```

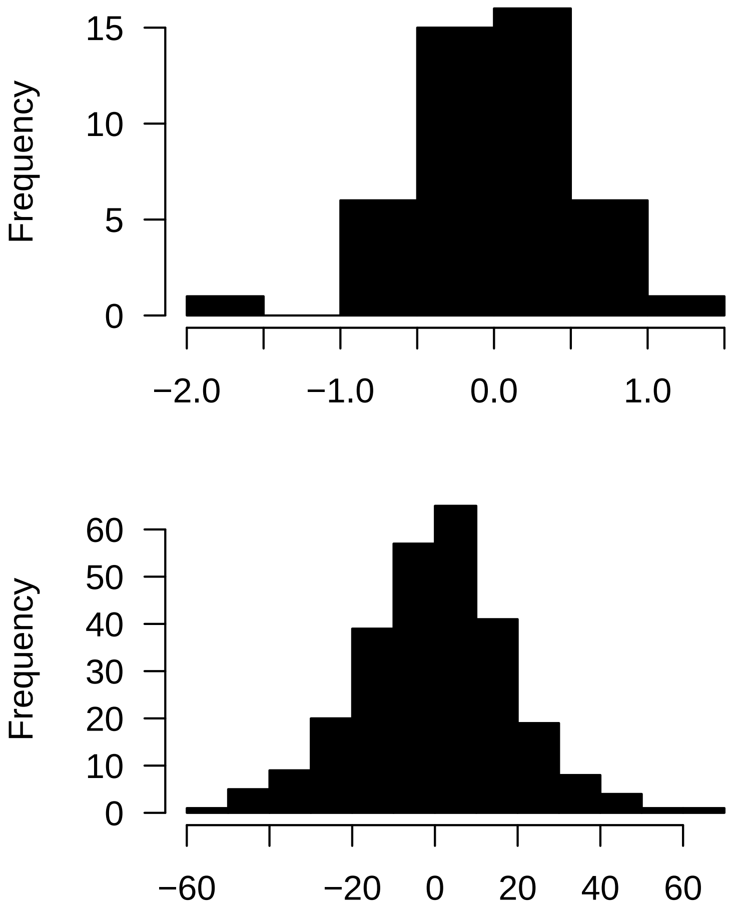


Let's now explore the distribution of the random effects and compare it with that of the residuals. The `random.effects` function returns the random effects. We can combine it with hist and we can see that random effects come from a normal distribution, as expected, with a different variance as that of the residuals (notice the different x scale).

```
head(random.effects(model))
```

```
##          Time
## 1  0.49678070
## 2  0.44133005
## 3 -0.05455666
## 4 -0.31447629
## 5  0.58834643
## 6 -1.55698489
```

```
par(mfrow = c(2, 1))
hist(random.effects(model)$Time, main = "", col = "black",
    xlab = "Random effects", las = 1)
hist(resid(model), main = "", col = "black", xlab = "Residuals",
    las = 1)
```



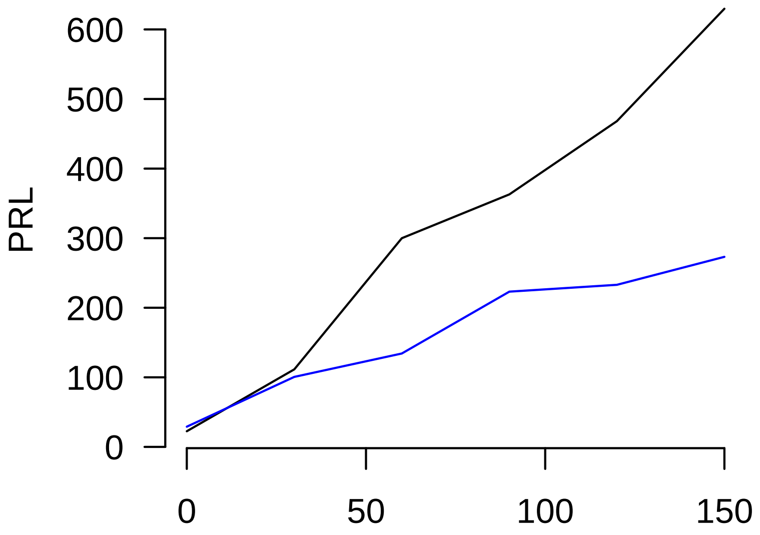
We can also check that R has correctly fitted a random slope model.

```
head(coef(model))
```

```
##   (Intercept) GroupBromo1 GroupBromo10     Time GroupBromo1:Time
## 1    17.23948   -3.546567     5.390355 3.963328        -1.924222
## 2    17.23948   -3.546567     5.390355 3.907877        -1.924222
## 3    17.23948   -3.546567     5.390355 3.411991        -1.924222
## 4    17.23948   -3.546567     5.390355 3.152071        -1.924222
## 5    17.23948   -3.546567     5.390355 4.054894        -1.924222
## 6    17.23948   -3.546567     5.390355 1.909562        -1.924222
##   GroupBromo10:Time
## 1         -3.244864
## 2         -3.244864
## 3         -3.244864
## 4         -3.244864
## 5         -3.244864
## 6         -3.244864
```

Note how all coefficients are the same for all the animals, but the slope for time has been changed for each animal. For instance, let's plot the data for mice 1 and 6. These are both control mice, but have quite different profiles.

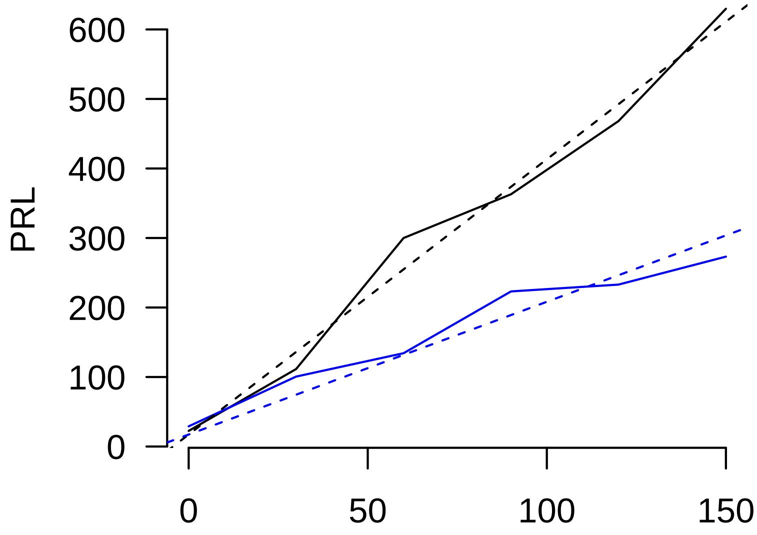
```
plot(PRL ~ Time, bromocriptine, subset = bromocriptine$Mouse ==
    1, t = "l", las = 1, bty = "n")
lines(PRL ~ Time, bromocriptine, subset = bromocriptine$Mouse ==
    6, col = "blue")
```



Indeed, looking at the fitted slopes, they are 3.96 and 1.91, while the intercept is 17.24 for both.

We can use abline to add those to the plot above.

```
abline(17.24, 3.96, lty = "dashed")  # Mouse 1
abline(17.24, 1.91, lty = "dashed")  # Mouse 6
```



Finally, let's use emmeans to compare the three different treatments. We use an extra argument here (cov.reduce = range) to tell emmeans to look at the range of the continuous variable time. If we do not use that, emmeans and pairs will make comparisons only at the mean time (that is, 75).

```
library(emmeans)

marginals <- emmeans(model, ~Group * Time, cov.reduce = range)
pairs(marginals, by = "Time")
```

```
## Time =   0:
##  contrast           estimate        SE df t.ratio p.value
##  CTRL - Bromo1      3.546567  5.451545 42   0.651  0.7931
##  CTRL - Bromo10    -5.390355  5.451545 42  -0.989  0.5879
##  Bromo1 - Bromo10  -8.936922  5.451545 42  -1.639  0.2407
##
## Time = 150:
##  contrast           estimate        SE df t.ratio p.value
##  CTRL - Bromo1    292.179869 30.559605 42   9.561  <.0001
##  CTRL - Bromo10   481.339196 30.559605 42  15.751  <.0001
##  Bromo1 - Bromo10 189.159327 30.559605 42   6.190  <.0001
##
## P value adjustment: tukey method for comparing a family of 3 estimates
```

Our post-hoc analysis has revealed that there is a significant difference between all of the groups at time 150, but not at time 0. Should you want to test all of the times you could use cov.reduce = unique.

A split plot experiment

Nested design

We are interested in the effect of soil composition and genotype on the growth of bamboo. You have identified three different alleles of the BoSus1 gene, which codes for the enzyme sucrose synthase.

You plant