**A repeated split-plot experiment with heteroscedastic errors**

Let’s imagine a field experiment, where different genotypes of khorasan wheat are to be compared under different nitrogen (N) fertilisation systems. Genotypes require bigger plots, with respect to fertilisation treatments and, therefore, the most convenient choice would be to lay-out the experiment as a split-plot, in a randomised complete block design. Genotypes would be randomly allocated to main plots, while fertilisation systems would be randomly allocated to sub-plots. As usual in agricultural research, the experiment should be repeated in different years, in order to explore the environmental variability of results.

What could we expect from such an experiment?

Please, look at the dataset ‘kamut.csv’, which is available on github. It provides the results for a split-plot experiment with 15 genotypes and 2 N fertilisation treatments, laid-out in three blocks and repeated in four years (360 observations, in all).

The dataset has five columns, the ‘Year’, the ‘Genotype’, the fertilisation level (‘N’), the ‘Block’ and the response variable, i.e. ‘Yield’. The fifteen genotypes are coded by using the letters from A to O, while the levels of the other independent variables are coded by using numbers. The following snippets loads the file and recodes the numerical independent variables into factors.

rm(list=ls())

dataset <- read.csv("<https://raw.githubusercontent.com/OnofriAndreaPG/agroBioData/master/kamut.csv>", header = T)

dataset$Block <- factor(dataset$Block)

dataset$Year <- factor(dataset$Year)

dataset$N <- factor(dataset$N)

head(dataset)

## Year Genotype N Block Yield

## 1 2004 A 1 1 2.235

## 2 2004 A 1 2 2.605

## 3 2004 A 1 3 2.323

## 4 2004 A 2 1 3.766

## 5 2004 A 2 2 4.094

## 6 2004 A 2 3 3.902

Additionally, it may be useful to code some ‘helper’ factors, to represent the blocks (within years) and the main-plots. The first factors (‘YearBlock’) has 12 levels (4 years and 3 blocks per year) and the second factor (‘MainPlot’) has 180 levels (4 years, 3 blocks per year and 15 genotypes per block).

dataset$YearBlock <- with(dataset, factor(Year:Block))

dataset$MainPlot <- with(dataset, factor(Year:Block:Genotype))

For the analyses, we will make use of the ‘plyr’ (Wickham, 2011), ‘car’ (Fox and Weisberg, 2011) and ‘nlme’ (Pinheiro et al., 2018) packages, which we load now.

library(plyr)

library(car)

library(nlme)

It is always useful to start by separately considering the results for each year. This gives us a feel for what happened in all experiments. What model do we have to fit to single-year split-plot data? In order to avoid mathematical notation, I will follow the notation proposed by Piepho (2003), by using the names of variables, as reported in the dataset. The treatment model for this split-plot design is:

Yield ~ Genotype \* N

All treatment effects are fixed. The block model, referencing all grouping structures, is:

Yield ~ Block + Block:MainPlot + Block:MainPlot:Subplot

The first element references the blocks, while the second element references the main-plots, to which the genotypes are randomly allocated (randomisation unit). The third element references the sub-plots, to which N treatments are randomly allocated (another randomisation unit); this latter element corresponds to the residual error and, therefore, it is fitted by default and needs not be explicitly included in the model. Main-plot and sub-plot effects need to be random, as they reference randomisation units (Piepho, 2003). The nature of the block effect is still under debate (Dixon, 2016), but I’ll take it as random (do not worry: I will also show how we can take it as fixed).

Coding a split-plot model in ‘lme’ is rather simple:

lme(Yield ~ Genotype \* N, random = ~1|Block/MainPlot

where the notation ‘Block/MainPlot’ is totally equivalent to ‘Block + Block:MainPlot’. Instead of manually fitting this model four times (one per year), we can ask R to do so by using the ‘ddply()’ function in the ‘plyr’ package. In the code below, I used this technique to retrieve the residual variance for each experiment.

lmmFits <- ddply(dataset, c("Year"),

function(df) summary( lme(Yield ~ Genotype \* N,

random = ~1|Block/MainPlot,

data = df))$sigma^2 )

lmmFits

## Year V1

## 1 2004 0.052761644

## 2 2005 0.001423833

## 3 2006 0.776028791

## 4 2007 0.817594477

We see great differences! The residual variance in 2005 is more that 500 times smaller than that observed in 2007. Clearly, if we pool the data and make an ANOVA, when we pool the data, we violate the homoscedasticity assumption. In general, this problem has an obvious solution: we can model the variance-covariance matrix of observations, allowing a different variance per year. In R, this is only possible by using the ‘lme()’ function (unless we want to use the ‘asreml-R’ package, which is not freeware, unfortunately). The question is: how do we code such a model?

First of all, let’s derive a correct mixed model. The treatment model is:

Yield ~ Genotype \* N

We have mentioned that the genotype and N effects are likely to be taken as fixed. The block model is:

~ Year + Year/Block + Year:Block:MainPlot + Year:Block:MainPlot:Subplot

The second element in the block model references the blocks within years, the second element references the main-plots, while the third element references the sub-plots and, as before, it is not needed. The year effect is likely to interact with both the treatment effects, so we need to add the following effects:

~ Year + Year:Genotype + Year:N + Year:Genotype:N

which is equivalent to writing:

~ Year\*Genotype\*N

The year effect can be taken as either as random or as fixed. In this post, we will show both approaches

**Year effect is fixed**

If we take the year effect as fixed and the block effect as random, we see that the random effects are nested (blocks within years and main-plots within blocks and within years). The function ‘lme()’ is specifically tailored to deal with nested random effects and, therefore, fitting the above model is rather easy. In the first snippet we fit a homoscedastic model:

modMix1 <- lme(Yield ~ Year \* Genotype \* N,

random = ~1|YearBlock/MainPlot,

data = dataset)

We could also fit this model with the ‘lme4’ package and the ‘lmer()’; however, we are not happy with this, because we have seen clear signs of heteroscedastic within-year errors. Thus, let’s account for such an heteroscedasticity, by using the ‘weights()’ argument and the ‘varIdent()’ variance structure:

modMix2 <- lme(Yield ~ Year \* Genotype \* N,

random = ~1|YearBlock/MainPlot,

data = dataset,

weights = varIdent(form = ~1|Year))

AIC(modMix1, modMix2)

## df AIC

## modMix1 123 856.6704

## modMix2 126 575.1967

Based on the Akaike Information Criterion, we see that the second model is better than the first one, which supports the idea of heteroscedastic residuals. From this moment on, the analyses proceeds as usual, e.g. by testing for fixed effects and comparing means, as necessary. Just a few words about testing for fixed effects: Wald F tests can be obtained by using the ‘anova()’ function, although I usually avoid this with ‘lme’ objects, as there is no reliable approximation to degrees of freedom. With ‘lme’ objects, I suggest using the ‘Anova()’ function in the ‘car’ package, which shows the results of Wald chi square tests.

Anova(modMix2)

## Analysis of Deviance Table (Type II tests)

##

## Response: Yield

## Chisq Df Pr(>Chisq)

## Year 51.072 3 4.722e-11 \*\*\*

## Genotype 543.499 14 < 2.2e-16 \*\*\*

## N 2289.523 1 < 2.2e-16 \*\*\*

## Year:Genotype 123.847 42 5.281e-10 \*\*\*

## Year:N 21.695 3 7.549e-05 \*\*\*

## Genotype:N 1356.179 14 < 2.2e-16 \*\*\*

## Year:Genotype:N 224.477 42 < 2.2e-16 \*\*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

One further aspect: do you prefer fixed blocks? Then you can fit the following model.

modMix4 <- lme(Yield ~ Year \* Genotype \* N + Year:Block,

random = ~1|MainPlot,

data = dataset,

weights = varIdent(form = ~1|Year))

**Year effect is random**

If we’d rather take the year effect as random, all the interactions therein are random as well (Year:Genotype, Year:N and Year:Genotype:N). Similarly, the block (within years) effect needs to be random. Therefore, we have several crossed random effects, which are not straightforward to code with ‘lme()’. First, I will show the code, second, I will comment it.

modMix5 <- lme(Yield ~ Genotype \* N,

random = list(Year = pdIdent(~1),

Year = pdIdent(~Block - 1),

Year = pdIdent(~MainPlot - 1),

Year = pdIdent(~Genotype - 1),

Year = pdIdent(~N - 1),

Genotype = pdIdent(~N - 1)),

data=dataset,

weights = varIdent(form = ~1|Year))

We see that random effects are coded using a named list; each component of this list is a *pdMat* object with name equal to a grouping factor. For example, the component ‘Year = pdIdent(~ 1)’ represents a random year effect, while ‘Year = pdIdent(~ Block – 1)’ represents a random year effect for each level of Block, i.e. a random ‘year x block’ interaction. This latter variance component is the same for all blocks (‘varIdent’), i.e. there is homoscedastic at this level.

It is important to remember that the grouping factors in the list are treated as nested; however, the grouping factor is only one (‘Year’), so that the nesting is irrelevant. The only exception is the genotype, which is regarded as nested within the year. As the consequence, the component ‘Genotype = pdIdent(~N – 1)’, specifies a random year:genotype effect for each level of N treatment, i.e. a random year:genotype:N interaction.

I agree, this is not straightforward to understand! If necessary, take a look at the good book of Gałecki and Burzykowski (2013). When fitting the above model, be patient; convergence may take a few seconds. I’d only like to reinforce the idea that, in case you need to test for fixed effects, you should not rely on the ‘anova()’ function, but you should prefer Wald chi square tests in the ‘Anova()’ function in the ‘car’ package.

Anova(modMix5, type = 2)

## Analysis of Deviance Table (Type II tests)

##

## Response: Yield

## Chisq Df Pr(>Chisq)

## Genotype 68.6430 14 3.395e-09 \*\*\*

## N 2.4682 1 0.1162

## Genotype:N 14.1153 14 0.4412

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Another note: coding random effects as a named list is always possible. For example ‘modMix2’ can also be coded as:

modMix2b <- lme(Yield ~ Year \* Genotype \* N,

random = list(YearBlock = ~ 1, MainPlot = ~ 1),

data = dataset,

weights = varIdent(form = ~1|Year))

Or, also as:

modMix2c <- lme(Yield ~ Year \* Genotype \* N,

random = list(YearBlock = pdIdent(~ 1), MainPlot = pdIdent(~ 1)),

data = dataset,

weights = varIdent(form = ~1|Year))

Hope this is useful! Have fun with it.