Let’s start from a real-life example.

# A split-plot experiment

The dataset ‘beet.csv’ is available in a web repository. It was obtained from a split-plot experiment with two experimental factors: three tillage methods (shallow ploughing, deep ploughing and minimum tillage) and two weed control methods (total and partial, meaning that the herbicide was sprayed broadcast or only along crop rows). Tillage methods were allocated to main-plots, while weed control methods were allocated to sub-plots and the experiment was designed in four complete blocks. A typical split-plot field experiment, indeed. The code below can be used to load the data.

```
library(tidyverse)
fileName <- "https://www.casaonofri.it/_datasets/beet.csv"
dataset <- read_csv(fileName)
dataset <- dataset %>%
  mutate(across(c(Tillage, WeedControl, Block), .fns = factor))
dataset
## # A tibble: 24 x 4
##    Tillage WeedControl Block Yield
##
##  1 MIN     TOT         1     11.6
##  2 MIN     TOT         2      9.28
##  3 MIN     TOT         3      7.02
##  4 MIN     TOT         4      8.02
##  5 MIN     PART        1      5.12
##  6 MIN     PART        2      4.31
##  7 MIN     PART        3      8.94
##  8 MIN     PART        4      5.62
##  9 SP      TOT         1     10.0
## 10 SP      TOT         2      8.69
## # … with 14 more rows
```

# The traditional approach

Split-plot designs are very commonly used in field experiments and they have been in fashion for (at least) eighty years, long before that the mixed model platform with REML estimation was largely available. Whoever has taken a course in ‘experimental design’ at the end of the 80s has studied how to perform a split-plot ANOVA by hand-calculations, based on the method of moments. For the youngest readers, it might be useful to give a few hints on what I used to do thirty years ago with the above dataset:

1. calculate the overall mean and the means for the levels of blocks, tillage, weed control and for the combined levels of tillage and weed control.
2. Calculate the means for the combined levels of blocks and tillage, which would correspond to the means for the twelve main-plots.
3. With all those means, calculate the deviances for all effects and interactions, as the sums of squared residuals with respect to the overall mean.
4. Derive the related variance, by using the appropriate number of degrees of freedom for each effect.
5. Calculate F ratios, based on the appropriate error stratum, i.e. the mean square for the

‘blocks × tillage’ combinations (so called: error A) and the residual mean square.

The most relevant aspect in the approach outlined above is the ‘block by tillage’ interaction; the mean square for this effect was used as the denominator in the F ratio, to test for the significance of the tillage main effect.

The above process was simple to teach and simple to grasp and I used to see it as a totally correct approach to balanced (orthogonal) split-plot data. Those of you who are experienced with SAS should probably remember that, before the advent of PROC MIXED in 1992, split-plot designs were analysed with PROC GLM, using the very same approach as outlined above.

Considering the above background, let’s see what I did when I switched to R?

# First step: ‘aov()’

Having the method of moments in mind, my first line of attack was to use the `aov()` function, as suggested in Venables and Ripley (2002) at pag. 283. Those authors make use of the nesting operator in the expression `Error(Block/Tillage)`.

```
mod.aov <- aov(Yield ~ Tillage*WeedControl +
                 Error(Block/Tillage), data = dataset)
summary(mod.aov)
##
## Error: Block
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals  3   3.66    1.22
##
## Error: Block:Tillage
##           Df Sum Sq Mean Sq F value Pr(>F)
## Tillage    2 23.656   11.83    19.4 0.0024 **
## Residuals  6  3.658    0.61
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Error: Within
##                      Df Sum Sq Mean Sq F value Pr(>F)
## WeedControl           1   3.32   3.320   1.225 0.2972
## Tillage:WeedControl   2  19.46   9.732   3.589 0.0714 .
## Residuals             9  24.40   2.711
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In the above definition, the block effect is regarded as random, while, in the traditional approach, it is often regarded as fixed. Indeed, still today, there is no consensus among agricultural scientists on whether the block effect should be regarded as random or fixed (see Dixon, 2016); for the sake of this exercise, let me regard it as fixed. After a few attempts, I discovered that I could move the effect of blocks from the `Error()` definition to the fixed effect formula and use the expression `Error(Block:Tillage)` to specify the uppermost error stratum.

```
mod.aov2 <- aov(Yield ~ Block + Tillage*WeedControl +
                 Error(Block:Tillage), data = dataset)
summary(mod.aov2)
##
```

```
## Error: Block:Tillage
##           Df Sum Sq Mean Sq F value Pr(>F)
## Block      3  3.660    1.22   2.001 0.2155
## Tillage    2 23.656   11.83  19.399 0.0024 **
## Residuals  6  3.658    0.61
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Error: Within
##                      Df Sum Sq Mean Sq F value Pr(>F)
## WeedControl           1   3.32   3.320   1.225 0.2972
## Tillage:WeedControl   2  19.46   9.732   3.589 0.0714 .
## Residuals             9  24.40   2.711
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Although the above code produces a warning message, the result is totally the same as I would have obtained by hand-calculations.

For me, the `aov()` function represented a safe harbour, mainly because the result was very much like what I would expect, considering my experience with mean squares and error strata. Unfortunately, I had to realise that there were several limitations to this approach and, finally, I had to switch to the mixed model platform.

# Second step: the mixed model framework

When making this switch to mixed models, I had the expectation that I should be able to reproduce the results obtained with the `aov()` function and, formerly, by hand-calculations.

I started with the `lme()` function in the 'nlme' package (Pinheiro et al., 2018) and I had the idea that I could simply replace the `Error(Block:Tillage)` statement with `random = ~1|Block:Tillage`. Unfortunately, using the `:` operator in the `lme()` function is not possible and I had to resort to using the nesting operator `'Block/Tillage'`. Consequently, I noted that the F test for the block effect was wrong. Not a very important problem, indeed, but I was so stupidly determined to reproduce my hand-calculations.

```
library(nlme)
mod.lme <- lme(Yield ~ Block + Tillage*WeedControl,
               random = ~1|Block/Tillage, data = dataset)
anova(mod.lme)
##                      numDF denDF   F-value p-value
## (Intercept)              1     9 120.85864  <.0001
## Block                    3     0   0.08045     NaN
## Tillage                  2     6   6.32281  0.0333
## WeedControl              1     9   1.77497  0.2155
## Tillage:WeedControl      2     9   5.20229  0.0315
```

Therefore, I tried to switch to the `lmer()` function in the 'lme4' package (Bates et al., 2015). With this platform, it was possible to include the 'block by tillage' interaction as a random effect, according to my usual workflow. Still, the results did not match to those obtained with the `aov()` function: an error message was raised and F ratios were totally different. Furthermore, p-levels were not even displayed (yes, now I know that we can use the 'lmerTest' package, but, please, wait a few seconds).

```
library(lme4)
mod.lmer.split <- lmer(Yield ~ Block + WeedControl*Tillage +
                     (1|Block:Tillage),
                     data=dataset)
anova(mod.lmer.split)
## Analysis of Variance Table
##                      npar  Sum Sq Mean Sq F value
## Block                   3  3.6596  1.2199  0.6521
## WeedControl             1  3.3205  3.3205  1.7750
## Tillage                 2 23.6565 11.8282  6.3228
## WeedControl:Tillage     2 19.4641  9.7321  5.2023
```

What's wrong with that? Why was I not able to reproduce my hand-calculations with the mixed model platform?

I investigated this matter and I found a very enlightening post by Douglas Bates (the author of 'nlme' and 'lme4'), which is available at this link. From there, it was clear to me that F ratios in mixed models are "*not based on expected mean squares and error strata*"; further ahead, it is said that there is "*a problem with the assumption that the reference distribution for these F statistics should be an F distribution with a known numerator of degrees of freedom but a variable denominator degrees of freedom*". In the end, it was clear to me that, according to Douglas Bates, the traditional approach of calculating p-values from F ratios based on expected mean squares and error strata was not necessarily correct.

I made some further research on this matter. Indeed, looking at the `aov()` output above, I noted that the residual mean square was equal to 2.711, while the mean square for the 'Block by Tillage' interaction was 0.6097. My beloved method of moments brought me to a negative estimate of the variance component for the 'block by tillage' interaction, that is $(0.6097 - 2.711)/4 = -0.5254$. I gasped: this was unreasonable and, at least, it would imply that the variance component for the 'block by tillage' random effect was not significantly different from zero. In other words, the mean square for the 'block by tillage' interaction and the mean square for the residuals were nothing but two separate estimates of the residual plot-to-plot error. I started being suspicious about my hand-calculations. Why did I use two estimates of the same amount as two different error strata?

I tried a different line of attack: considering that the 'block by tillage' interaction was not significant, I removed it from the model. Afterwards I fitted a linear fixed effect model, where the two error strata had been pooled into the residual error term. I obtained the very same F ratios as those obtained from the 'lmer' fit.

```
mod.lm <- lm(Yield ~ Block + WeedControl*Tillage, data=dataset)
anova(mod.lm)
## Analysis of Variance Table
##
## Response: Yield
##                      Df  Sum Sq Mean Sq F value  Pr(>F)
## Block                 3  3.6596  1.2199  0.6521 0.59389
## WeedControl           1  3.3205  3.3205  1.7750 0.20266
## Tillage               2 23.6565 11.8282  6.3228 0.01020 *
## WeedControl:Tillage   2 19.4641  9.7321  5.2023 0.01922 *
## Residuals            15 28.0609  1.8707
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In that precise moment when I noted such a result, it was clear to me that, even with simple and orthogonal split-plot designs, hand-calculations do not necessarily produce correct results and should never, ever be used as the reference to assess the validity of a mixed model fit.

# Suggestions for dinosaurs

If you are one of those who have never taken a lesson about expected mean squares and error strata, well, believe me, you are lucky! For us dinosaurs, switching to the mixed model platform may be a daunting task. We need to free up our minds and change our workflow; a few suggestions are following.

# Rule 1: change model building process

In principle, do no insist on including the 'block by tillage' interaction in the model. With split-plot experiments, the main-plot is to be regarded as a *grouping structure*, wherein we take repeated measures in different sub-plots. These measures are correlated, as they are more alike than measures taken in different sub-plots.

Therefore, for this grouping structure (and for all grouping structures in general) we need to code a *grouping factor*, to uniquely identify the repeated measures in each main-plot. This factor must be included in the model, otherwise we violate the basic assumption of independence of model residuals. Consider that the main-plot represent the randomisation units to which the tillage treatments were allocated; therefore, the main plot factor needs to be included in the model as a random effect. Please refer to the good paper of Piepho et al. (2003) for further information on this model building approach.

In the box below I created the main-plot factor by using `dplyr()` to combine the levels of blocks and tillage methods. The difference with the traditional approach of using the 'block by tillage' interaction in the model is subtle, but, in this case, the `lme()` function returns no error. Please, note that, having no interest in the estimation of variance components, I have fitted this model by maximum likelihood estimation: it is confirmed that the main-plot random effect is zero (see the output of the `VarCorr()` function).

```
dataset <- dataset %>%
  mutate(mainPlots = factor(Block:Tillage))
mod.lme2 <- lme(Yield ~ Block + Tillage * WeedControl,
                random = ~1|mainPlots, data = dataset,
                method = "ML")
VarCorr(mod.lme2)
## mainPlots = pdLogChol(1)
##             Variance     StdDev
## (Intercept) 4.462849e-10 2.112546e-05
## Residual    1.169203e+00 1.081297e+00
```

# Rule 2: change the approach to hypotheses testing

In the agricultural sciences we have been very much familiar with ANOVA tables, showing all fixed effects along with their significance level. I am very much convinced that we should refrain from such a (possibly bad) habit. Indeed, there is no point in testing the significance of main effects before testing the significance of the 'tillage by weed control' interaction, as main effects are marginal to the interaction effect.

At first, we need to concentrate on the interaction effect. According to maximum likelihood theory, it is very logic to think of a Likelihood Ratio Test (LRT), which consists of comparing the likelihoods of two alternative and nested models. In this case, the model above ('mod.lme2') can be compared with a 'reduced' model without the 'tillage by weed control' interaction term: if the two likelihoods are similar, that would be a sign that the interaction effect is not significant. The reduced model fit is shown below.

```
mod.lme3 <- lme(Yield ~ Block + Tillage + WeedControl,
                random = ~1|mainPlots, data = dataset,
                method = "ML")
```

The logarithms of the two likelihoods show that the 'full model' (with the interaction term) is more 'likely' than the reduced model. The LRT is calculated as twice the difference between the two log-likelihoods (the logarithm of the ratio of two numbers is the difference of the logarithms, remember?).

```
ll2 <- logLik(mod.lme2)
ll3 <- logLik(mod.lme3)
ll2; ll3
## 'log Lik.' -35.93039 (df=11)
## 'log Lik.' -42.25294 (df=9)
LRT <- - 2 * (as.numeric(ll3) - as.numeric(ll2))
LRT
## [1] 12.6451
```

For large samples and under the null hypothesis that the two models are not significantly different, the LRT is distributed according to a $\chi^2$ with two degrees of freedom (i.e. the difference in the number of model parameters used by the two models). We could use such an assumption to obtain a p-level for the null, for example by way of the `anova()` function, to which we pass the two model objects as arguments.

```
anova(mod.lme2, mod.lme3)
##          Model df       AIC      BIC    logLik   Test L.Ratio
p-value
## mod.lme2     1 11  93.86078 106.8194 -35.93039
## mod.lme3     2  9 102.50589 113.1084 -42.25294 1 vs 2 12.6451
0.0018
```

However, our experiment consists of only 24 observations and the large sample theory should not hold. Therefore, instead of relying on the $\chi^2$ distribution, we can build an empirical sampling distribution for the LRT with Monte Carlo simulation (parametric bootstrap). The process is as follows:

1. simulate a new dataset under the reduced model, using the fitted parameter estimates and assuming normality for the errors and random effects;
2. fit to this dataset both the full and the reduced model;
3. compute the LRT statistic;
4. repeat steps 1 to 3 many times (e.g., 10000);
5. examine the distribution of the bootstrapped LRT values and compute the proportion of those exceeding 12.6451 (empirical p-value).

To this aim, we can use the `simulate()` function in the 'nlme' package. We pass the reduced model object as the first argument, the full model as the argument 'm2', the number of simulations and the seed (if we intend to obtain reproducible results). The fit may take quite a

few minutes.

```
y <- simulate(mod.lme3, nsim = 10000, m2 = mod.lme2, method="ML",
              set.seed = 1234)
lrtSimT <- as.numeric(2*(y$alt$ML[,2] - y$null$ML[,2]))
length(lrtSimT[lrtSimT > 12.6451])/length(lrtSimT)
## [1] 0.0223
```

We conclude that the interaction is significant and we can go ahead with further analyses.

# Take-home message

What is the take-home message for this post? When we have to analyse a dataset coming from a split-plot experiment, R forces us to use the mixed model platform. We should not necessarily expect to reproduce the approach and the results we were used to obtain when we made our hand-calculations based on least squares and the method of moments. On the contrary, we should adapt our model building and hypothesis testing process to such a very powerful platform, wherein the slit-plot is treated on equal footing to all other types of repeated measures designs.

Hope this was fun! If you have any comments, drop me a line to the email below.