

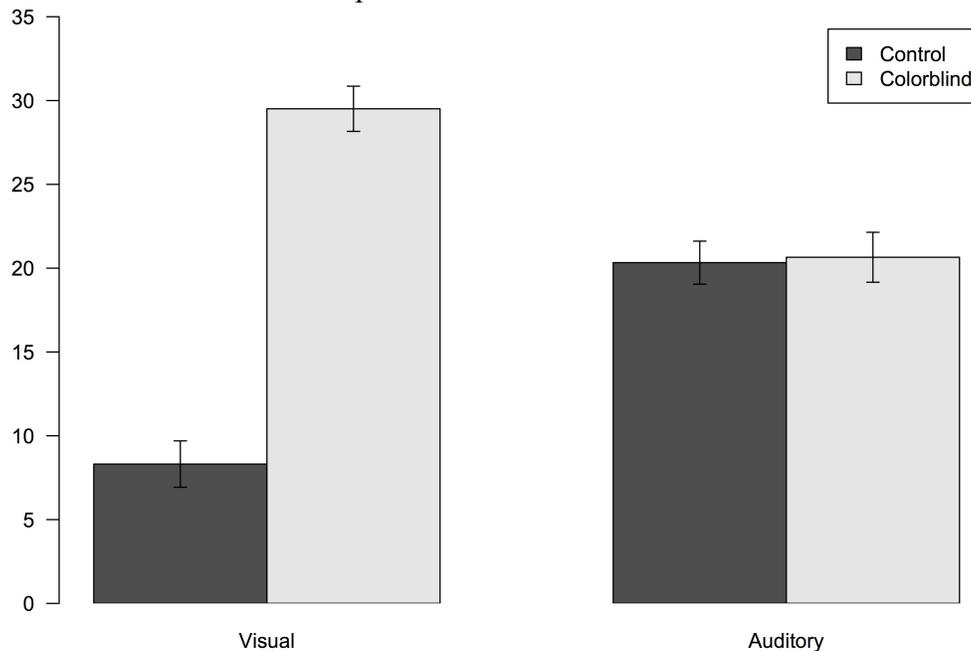
# A few useful tips for LME4 (R package)

Alexandre Cremers

## Re-encode your contrasts!

By default, R uses the 'treatment coding'. This means that when you run a linear model (lm or lmer), the first level of a factor is taken as the baseline. The aov function for ANOVAs automatically re-encodes your factors to sum-contrasts (deviation coding), hence it uses the mean across the different levels as its baseline, which is usually more meaningful than the first level.

Let us see what this means with an example:



These are artificial data for the following (thought) experiment: participants had to identify a color by clicking on its name after either seeing a patch of color (visual) or hearing the name of the color (auditory). We compare colorblind participants to a control group with normal vision, and the dependent variable is some transformation of their response times.

```
Anova: summary(aov(RT~SubjType*Modality))
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
SubjType	1	1735.8	1735.8	60.955	1.61e-10 ***
Modality	1	37.4	37.4	1.314	0.257
SubjType:Modality	1	1634.1	1634.1	57.386	3.88e-10 ***
Residuals	56	1594.7	28.5		

```
Linear model with default parameters: summary(lm(RT~SubjType*Modality))
```

Estimate	SE	t value	Pr(> t )
----------	----	---------	----------

(Intercept)	8.314	1.378	6.034	1.34e-07 ***
SubjTypeCB	21.195	1.949	10.877	1.99e-15 ***
ModalityAud	12.017	1.949	6.167	8.12e-08 ***
SubjTypeCB:ModalityAud	-20.875	2.756	-7.575	3.88e-10 ***

Problem: the default behavior of the `lm` function is to evaluate the effect of `SubjType` within the Visual modality, and the effect of `Modality` among control subjects. As you can see, the intercept corresponds to control participants in the visual modality, not to the grand mean.

Because of this, the model returns a significant effect of modality. If you were to change the ordering of the levels of the factors, you would get different results and this makes the interpretation very complicated.

We can force `lm` to behave like `aov` by re-encoding the contrasts.

```
contrasts(Data$SubjType)<-contr.sum(2) # 2 stands for the number of levels in the factor
contrasts(Data$Modality)<-contr.sum(2)
```

`lm` will now evaluate the average effects of `Modality` and `SubjType`:

	Estimate	SE	t value	Pr(> t )
(Intercept)	19.7010	0.6889	28.597	< 2e-16 ***
SubjType1	-5.3786	0.6889	-7.807	1.61e-10 ***
Modality1	-0.7896	0.6889	-1.146	0.257
SubjType1:Modality1	-5.2188	0.6889	-7.575	3.88e-10 ***

Note that the *p*-values are now exactly the same as in the ANOVA. Be careful if you want to interpret the estimates. First, you lost the names of your levels. Second, you have to multiply the estimates by 2 in order to get the estimated difference between the two levels. Indeed, the estimate for `Modality1` corresponds to the difference between the grand mean and the visual modality. Going from the auditory to the visual modality now represents a move from the value -1 to +1 on the modality "scale". You can avoid this by encoding your levels as -0.5 and +0.5, for instance writing "`contr.sum(2)/2`" in the previous formula.

## Compare different models!

By default, `lmer` returns *t*-values (but no *p*-values), and `glmer` returns *z*-values and *p*-values. Assuming you respected the assumptions of `lmer` (e.g., normally distributed random effects) and have enough participants and items, you can more or less treat the *t*-values as *z*-values, from which you can extract *p*-values (cf slides). On the other hand, the *z*-values of `glmer` (and the associated *p*-values) are not reliable. In any case, the best way to get a reliable *p*-value is through model comparison.

Example: Imagine we run a mixed-effect model on the previous experiment. `Modality` is within subject, but not `SubjType`, so we should run the following model:

```
myModel<-lmer(Response~Modality*SubjType + (1+Modality|Subject), data=mydata)
```

If we want to assess the global significance of our model, we can compare it to the null model (in which we remove all fixed effects but the intercept).

```
nullmodel<-lmer(Response~1 + (1+Modality|Subject), data=mydata)
anova(nullmodel,mymodel)      # Ignore the name, this function can do more than anova's.
```

If we want to test specifically for the effect of the interaction, we have to compare the model to a version without interaction (the following two expressions are equivalent):

```
nointmodel<-lmer(Response~Modality+SubjType + (1+Modality|Subject), data=mydata)
nointmodel<-lmer(Response~Modality*SubjType - Modality:SubjType + (1+Modality|
Subject), data=mydata)
anova(nointmodel,mymodel)
```

Unfortunately, there is no simple way to test for a main effect independently of the interaction. Imagine we want to evaluate the main effect of SubjType. We could try one of the following:

```
noFEmodel<-lmer(Response~Modality+Modality:SubjType + (1+Modality|Subject),
data=mydata)
noFEmodel<-lmer(Response~Modality*SubjType - SubjType + (1+Modality|Subject),
data=mydata)
```

Unfortunately, it will not work (R will re-encode the factors in a strange way and noFEmodel will end up being exactly equivalent to mymodel). The simplest solution is to compare reduced models:

```
model1<-lmer(Response~Modality+SubjType + (1+Modality|Subject), data=mydata)
model2<-lmer(Response~Modality + (1+Modality|Subject), data=mydata)
anova(model1, model2)
```

If you really want to keep the interaction in the two models, you can use the following trick in order to define a factor encoding your interaction (but R will not know it corresponds to an interaction):

```
mydata$Interaction<-contrasts(mydata$Modality)[mydata$Modality]*
                    contrasts(mydata$SubjType)[mydata$SubjType]
```

Then you can compare the following two models:

```
fullmod<-lm(Response~Modality+SubjType+Interaction + (1+Modality|Subject),
data=mydata)
partmod<-lm(Response~Modality+Interaction + (1+Modality|Subject), data=mydata)
anova(partmod,fullmod)
```

fullmod will correspond to the original mymodel. partmod will be the model with interaction but no main effect of SubjType. This works well with an interaction between factors with only 2 levels each. For interaction between factors with more levels, you will need to define more interaction factors and this may become slightly more complicated...

Reference:

Levy, R. (2014). Using R formulae to test for main effects in the presence of higher-order interactions. *arXiv preprint arXiv:1405.2094*.