Multilevel modeling worksheet: Exercise 2

Getting accurate inferences for multilevel models

N.B: the discussion here focuses on R, but the challenge of getting accurate inferences applies to all software that fits multilevel models. The main difference is that many packages apply a default solution without explaining the limitations of that approach.

Q4 The *t* statistics and *F* statistics from the summary function are exact only in certain very restricted cases (effectively only for a balanced repeated measures ANOVA). To obtain inferences one can use a likelihood ratio test (LRT) or profiling of likelihood (for confidence intervals). These are thought to be better than using *t* or *F* where the correct *df* are not known (and where there may be boundary effects), but still far from ideal.

For further details see: http://glmm.wikidot.com/faq

To get profile likelihood confidence intervals use the function `confint(model1)` and to get likelihood ratio tests use anova(model1, model2) to compare two models or `drop1(model1)` for a single model (which drops the highest order effects one at a time).

a) *What is the 95% CI (from profiling) for the effect of* attract*?*

b) *What is LRT for the effect of* attract*?*

c) *What is LRT for the fixed effect nested model versus the null nested model from above?*

N.B. Note that tests of fixed effects require that the model be fitted as full maximum likelihood (ML = TRUE). Recent versions of lme4 automatically refit the model full maximum likelihood rather than restricted maximum likelihood when anova() is used to compare models. Look out for this in the output.

Q5 The approach for obtaining inferences regarded as safest and best is to use MCMC estimation. There are many ways to do this in R, but one of the easiest (for fairly simple models) is to use the `MCMCglmm` package.[1]

Start by running a small number of iterations to see if everything is OK.

```
library(MCMCglmm)
niterations <- 10000
pitch3.mcmc <- MCMCglmm(pitch ~ base + attract, random= ~ Participant,
 nitt=niterations, data=pitch.dat)
summary(pitch3.mcmc)
```

a) Are there any clues in the model summary that the MCMC estimates have converged?

b) Plot the MCMC trace using `plot(pitch3.mcmc)` and hitting return to cycle through the graphs. A healthy trace should look like a fat, hairy caterpillar. *What you think?*

c) Re-run the model with 50000 iterations. *How do the plots look now?*

d) *What is the 95% HPD interval for the effect of* attract*? How does it compare with the CI from profiling?*

Optional

This part is probably best done in your own time. Packages such as SAS use a correction to deal with the *df* concerns for multilevel models. These are available in R using the `pbkrtest` package, but I suggest using the convenience functions in the lmerTest package which call `pbkrtest` for you. Install the packages if you don't have them:

[1] Install it within R Studio or use install.packages('MCMCglmm') from the R console.

```
install.packages("pbkrtest")
install.packages("lmerTest")
```

Then load the package and refit the model you want to test. (It must be re-fit you can't just use the old model object). For example:

```
library(lmerTest)

pitch3.fe.refit <- lmer(pitch ~ base + attract + (1|Participant/Face),
  data=pitch.dat), ddf="Kenward-Roger")
anova(pitch3.fe.refit, ddf="Kenward-Roger")
```

Although this approach is known to work well for multilevel normal response models with nested structures, it may break down in other situations (and it also doesn't provide confidence intervals.)

Comparing confidence interval methods.

An interesting additional exercise is to compare CIs produced by different methods. The latest versions of `lme4` allow this very easily. Particularly useful is to compare the Wald (a fairly common normal approximation) to the profile likelihood and parametric bootstrapping methods.

Warning! Parametric bootstrapping can take a long time (depending on the number of simulations) and may not converge. For example, 5000 simulations was too much for my fairly powerful desktop machine.

```
confint(pitch3.fe, method="profile")
confint(pitch3.fe, method="Wald")
confint(pitch3.fe, method="boot", nsim=500)
confint(pitch3.fe, method="boot", nsim=1000)
```