Mixed models

(Genealized) Linear Mixed Models (GLMM)

Mixed models are used extremely often in psychology to account for dependence structures in the data and they offer a more general framework that subsume specific models, such as within-subject ANOVA.

We will use package **simr** (Green et al., 2016), which is in turn based on the the most widespread package for GLMM, **lme4** (Bates, 2015).

We will focus on power for detecting fixed effects, which I assume to be by far the most widespread issue (see Green et al., 2016 or type *?powerSim* for additional possibilities).

p-values in mixed models with lmerTest

- A crucial ingredient of power analysis is a statistical test to reject H0.
- However, lme4 does not return p-values, and for a good reason: Degrees of freedom cannot be computed in mixed models (see this famous post by Bates, https://stat.ethz.ch/pipermail/r-help/2006-May/094765.html)
- Package *lmerTest* is a useful companion to lme4, as it includes p-values back in lme4 using the Satterthwaite degrees of freedom approximation (Kuznetsova et al., 2017).
- Since Satterthwaite approximation is quite widespread has been shown to perform satisfactorily (Luke, 2017), I will show how to implement power considering lmerTest's p < .05 this as our statistical test.

Simon task dataset

Another important ingrendient of power analysis is a **specification of model parameters under H1**. This can be very complex in mixed models: The easiest way to use package *simr* is by having available a dataset (e.g., a previous study, a pretest) and by varying some parameters of interest (see Green et al., 2016 for how to specify a model from scratch).

In our examples, we will use a dataset on Simon Task including 16 subjects each one performing 1664 trials.

```
load("data/simon.RData")
```

The dataset was kindly provided by dott. Carlotta Lega and prof. Luigi Cattaneo (University of Verona). Variables and experimental conditions that were not relevant for this example were removed. One subject (#1) did not complete all conditions and was also removed.

Simon task dataset (1)

In Simon task, subjects discriminate the color of a stimulus (red vs. green) by responding with the left vs. right key. Half subjects responded to red with the right hand and half with the left hand. Participants have to ignore the location in which the stimuli are presented, which could be left vs. right. The Simon effect is the facilitation (faster RTs and higher accuracy) if the stimulus is presented in the same location of the response (congruent condition) compared to when it is presented in the opposite location (incongruent condition), even if stimulus location is irrelevant to the task.

Variables included in the dataset are:

- subject: identifier
- target_side: whether the target was presented on the left (0) vs. right (1)
- response_side: whether the correct response is on the left (0) vs. right (1)
- congruency: whether trial is incongruent (0) vs. congruent (1)
- target_color: the color of the target, green (0) vs. red (1)
- RT: response times
- accuracy: incorrect (0) vs. correct (1)

Simon task dataset (2)

```
summary(simon)
```

```
    subject      target_side   response_side congruency
target_color
 2      : 1664   left :13312   left :13312   0:13312    green:13312
 3      : 1664   right:13312   right:13312   1:13312    red  :13312
 4      : 1664
 5      : 1664
 6      : 1664
 7      : 1664
 (Other):16640
       RT               accuracy
 Min.   :   57.0   Min.   :0.0000
 1st Qu.:  275.0   1st Qu.:1.0000
 Median :  340.0   Median :1.0000
 Mean   :  383.2   Mean   :0.9633
 3rd Qu.:  425.0   3rd Qu.:1.0000
 Max.   :23957.0   Max.   :1.0000
```

Simon task dataset (3)

The data summary shows immediately that we have some extreme RTs, which we should handle. For the sake of this example, we simply remove all RTs > 1500ms, which are only 0.44%

```
mean(simon$RT > 1500)
```

```
[1] 0.004356971
```

```
simon <- filter(simon, RT <= 1500)
```

Since RT data are generally not normally distributed, one might log-transform RTs or consider more refined approaches (e.g., see Lo & Andrews, 2015). In this example, we will just ignore deviations from normality.

Let’s predict RTs using a mixed model

We want to see whether the Simon effect affects RTs. To do so, we predict *RT* from *congruency*, *target_color* and their interaction, the Simon effect being given by the main effect of congruency. Since RTs are nested within subjects, we need to include at least a random intercept by subject, which takes into account the fact that different individuals may vary in their average speed.

In R, this linear mixed model can be fitted using function **lmer** in package *lme4*

```
model1 <- lmer(RT ~ congruency*target_color + (1|subject),
            data = simon)
```

What we show here about power holds also if random slopes are added (but computations become slower).

Let's predict RTs using a mixed model (1)

```r
summary(model1)
```

```
Linear mixed model fit by REML. t-tests use Satterthwaite's method [
lmerModLmerTest]
Formula: RT ~ congruency * target_color + (1 | subject)
   Data: simon

REML criterion at convergence: 342328.3

Scaled residuals: 
    Min      1Q  Median      3Q     Max 
-2.8347 -0.5327 -0.1914  0.2365  7.6608 

Random effects:
 Groups   Name        Variance Std.Dev.
 subject  (Intercept)  4470     66.85  
 Residual             23710    153.98  
Number of obs: 26508, groups:  subject, 16

Fixed effects:
                             Estimate Std. Error        df t value Pr(>|t|)    
(Intercept)                   390.777     16.820    15.288  23.232 2.42e-13 ***
congruency1                   -26.523      2.674 26489.000  -9.917  < 2e-16 ***
target_colorred                -5.897      2.675 26489.001  -2.204   0.0275 *  
congruency1:target_colorred     1.566      3.783 26489.000   0.414   0.6788    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Correlation of Fixed Effects:
            (Intr) cngrn1 trgt_c
congruency1 -0.080              
targt_clrrd -0.079  0.500       
cngrncy1:t_  0.056 -0.707 -0.707
```

Let’s predict RTs using a mixed model (2)

Simon effect turns out to be significant: A facilitation of around 27ms is observed for congruent trials (p < .001). There is also a small facilitation (-6ms) for target_color, with faster RTs in when the target is red, but no interaction effects.

Power with simr (1)

We want to know the power that we had to detect a Simon effect (i.e., main effect of congruency) as strong as the one that we have just observed, with the current setting (number of individuals and of trials). In package *simr*, this is simply achieved with the function **powerSim**.

- The first argument is a fitted *lme4* object, in our case **model1**
- The second argument, **test**, specifies the effect that we wish to test. In our case, we wish to test a fixed effect, therefore we specify **fixed()**.
- Within fixed, we specify that the effect of interest is called **"congruency1"** in model1
- we also specify **method = "t"**, which means that we want to use a t-test with Satterthwaite approximation of df. To see more options, you can type *?powerSim* and *?tests*
- **nsim** specifies the number of simulations.
- **progress = FALSE** only tells R not to plot a progress bar.

```
ps1 <- powerSim(model1,
                test = fixed("congruency1", method = "t"),
                nsim = 1000,
                progress = FALSE)
```

Power with simr (2)

R warns us that this is a post-hoc power analysis, which should never be used to justify one's current sample size as adequate. The output suggests that we have high power to observe the effect with the current setting.

```
ps1
```

```
Power for predictor 'congruency1', (95% confidence interval):
      100.0% (99.63, 100.0)

Test: t-test with Satterthwaite degrees of freedom (package
lmerTest)
      Effect size for congruency1 is -27.

Based on 1000 simulations, (0 warnings, 0 errors)
alpha = 0.05, nrow = 26508

Time elapsed: 0 h 9 m 57 s

nb: result might be an observed power calculation
```

Power with simr (3)

simr allows also to vary several parameters, to obtain a custom specification of the model parameters under H1, as well as a different sample size or number of trials.

For example, we might be interested in inspecting whether we would have sufficient power also if the facilitation effect was of only 10 ms instead of 27. In simr, we can achieve this by specifying a different effect for the fixed effect using command **fixef** and re-running the simulation.

```
model2 <- model1
fixef(model2)["congruency1"] <- 10
```

```
ps2 <- powerSim(model2,
                test = fixed("congruency1", method = "t"),
                nsim = 1000,
                progress = FALSE)
```

Power with simr (4)

Results of the simulation suggest that 16 subjects would be sufficient to detect a facilitation of only 10s, everything else being equal.

```
ps2
```

```
Power for predictor 'congruency1', (95% confidence interval):
      96.90% (95.63, 97.88)

Test: t-test with Satterthwaite degrees of freedom (package
lmerTest)
      Effect size for congruency1 is 10.

Based on 1000 simulations, (0 warnings, 0 errors)
alpha = 0.05, nrow = 26508

Time elapsed: 0 h 9 m 26 s
```

Power with simr (5)

We might want to inspect what power we would have if the sample size was 30 instead of 16. To do this, we might use command **extend**. This command changes the levels of a variable in the model specified by **along** to a number of levels specified by **n**.

```r
model3 <- extend(model2, along = "subject", n = 30)
```

We can then simply run a power simulation on the new model

```r
ps3 <- powerSim(model3,
                test = fixed("congruency1", method = "t"),
                nsim = 1000,
                progress = FALSE)
```

Power with simr (5)

The results suggest that the power is extremely high

```
ps3
```

```
Power for predictor 'congruency1', (95% confidence interval):
      100.0% (99.63, 100.0)

Test: t-test with Satterthwaite degrees of freedom (package
lmerTest)
      Effect size for congruency1 is 10.

Based on 1000 simulations, (0 warnings, 0 errors)
alpha = 0.05, nrow = 49699

Time elapsed: 0 h 12 m 44 s
```

Power with simr (6)

We can also use command **extend** to change the number of observations within clusters (in this case, the number of trials within each subject). For example we might want to inspect the power that we would obtain if we had only 50 trials by subject. In this case, instead of *along*, we can specify argument **within**.

```r
model4 <- extend(model3, within = "subject", n = 50)

ps4 <- powerSim(model4,
                test = fixed("congruency1", method = "t"),
                nsim = 1000,
                progress = FALSE)
```

Power with simr (6)

In this case, power does not seem to be particularly satisfactory, even with 30 participants.

```
ps4
```

```
Power for predictor 'congruency1', (95% confidence interval):
      14.50% (12.37, 16.84)

Test: t-test with Satterthwaite degrees of freedom (package
lmerTest)
      Effect size for congruency1 is 10.

Based on 1000 simulations, (3 warnings, 0 errors)
alpha = 0.05, nrow = 1500

Time elapsed: 0 h 1 m 26 s
```

Power with simr (8)

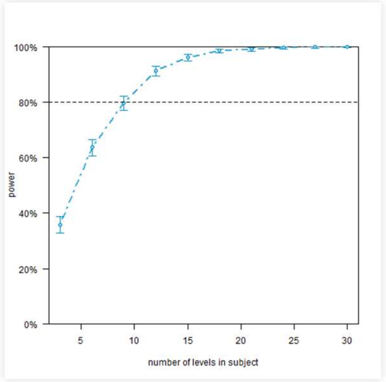
Finally, function **powerCurve** allows inspecting how power varies as a function of a set of values for a parameter. For example, if we want to inspect how power varies as a function of sample size in our model3 (10ms facilitation on max 30 subjects), we can use the following code.

```r
pc <- powerCurve(model3,
                  along = "subject",
                  test = fixed("congruency1", method = "t"),
                  nsim = 1000,
                  progress = FALSE)
```

Power with simr (9)

Results suggest that a sample if size > 10 would be sufficient to get 80% power

```
plot(pc)
```



Power with simr (10)

simr is not restricted to linear models. Everything we have seen so far applies also to generalized models, for example logistic regression.

The following code uses fits a model similar to the one that we have just inspected to predict **accuracy**. Since accuracy is a binary variable, we need to specify a logistic mixed model, which can be achieved with command **glmer** by specifying *family = "binomial"*

```
model5 <- glmer(accuracy ~ congruency*target_color + (1|subject),
                family = "binomial",
                data = simon)
```

Power with simr (11)

```r
summary(model5)
```

```
Generalized linear mixed model fit by maximum likelihood (Laplace
  Approximation) [glmerMod]
 Family: binomial  ( logit )
Formula: accuracy ~ congruency * target_color + (1 | subject)
   Data: simon

     AIC      BIC   logLik deviance df.resid
  7846.6   7887.5  -3918.3   7836.6    26503

Scaled residuals:
     Min       1Q   Median       3Q      Max
-15.5381   0.1207   0.1707   0.2075   0.3843

Random effects:
 Groups  Name        Variance Std.Dev.
 subject (Intercept) 0.5874   0.7664
Number of obs: 26508, groups:  subject, 16

Fixed effects:
                            Estimate Std. Error z value Pr(>|z|)
(Intercept)                  3.19632    0.20162  15.853  < 2e-16 ***
congruency1                  0.75734    0.09822   7.711 1.25e-14 ***
target_colorred              0.05957    0.08174   0.729    0.466
congruency1:target_colorred -0.13631    0.13800  -0.988    0.323
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Correlation of Fixed Effects:
            (Intr) cngrn1 trgt_c
congruency1 -0.162
targt_clrrd -0.197  0.404
cngrncy1:t_  0.116 -0.711 -0.592
```

Power with simr (12)

With GLMM simr works in the usual way. This time, we don't specify that we want to use a t-test, but we leave the default (z test). Furthermore, since glmer takes longer to run, we turn down the number of simulated datasets to 100 (in real setting, use nsim >= 1000 if possible)

```r
ps5 <- powerSim(model5,
                test = fixed("congruency1"),
                nsim = 100,
                progress = FALSE)
```

Power with simr (11)

Here we can collect the results

```
ps5
```

```
Power for predictor 'congruency1', (95% confidence interval):
      100.0% (96.38, 100.0)

Test: z-test
      Effect size for congruency1 is 0.76

Based on 100 simulations, (0 warnings, 0 errors)
alpha = 0.05, nrow = 26508

Time elapsed: 0 h 9 m 47 s

nb: result might be an observed power calculation
```

Conclusions

In conclusion

- In many cases, analytic solutions do exist and work quite well. If you can, use them.
- Sometimes they do not work, and simulation is the only way to go. Be patient, because in some cases you might need to program quite a bit and to wait even longer to collect your results.
- All packages I presented include a much larger variety of functions than those that I have showd you. The R help files are often extremely informative. Use them!
- I don't expect that you will be able to use immediately all the methods we have seen today, but if you need to perform power analysis I hope that these slides as well as the references within the slides might turn out to be useful.

Thank you all for your attention and good luck with your research!

References - suggested readings in bold (1)

Baron, R. M., & Kenny, D. A. (1986). The moderator-mediator variable distinction in social psychological research: Conceptual, strategic, and statistical considerations. Journal of Personality and Social Psychology, 51(6), 1173-1182. https://doi.org/10.1037/0022-3514.51.6.1173

Bates, D., Machler, M., Bolker, B., & Walker, S. (2015). Fitting Linear Mixed-Effects Models Using lme4. Journal of Statistical Software, 67(1). https://doi.org/10.18637/jss.v067.i01

Green, P., & MacLeod, C. J. (2016). SIMR: an R package for power analysis of generalized linear mixed models by simulation. Methods in Ecology and Evolution, 7(4), 493-498. https://doi.org/10.1111/2041-210X.12504

Kuznetsova, A., Brockhoff, P. B., & Christensen, R. H. B. (2017). lmerTest Package: Tests in Linear Mixed Effects Models. Journal of Statistical Software, 82(13). https://doi.org/10.18637/jss.v082.i13

Lo, S., & Andrews, S. (2015). To transform or not to transform: using generalized linear mixed models to analyse reaction time data. Frontiers in Psychology, 6, 1-16. https://doi.org/10.3389/fpsyg.2015.01171

Luke, S. G. (2017). Evaluating significance in linear mixed-effects models in R. Behavior Research Methods, 49(4), 1494-1502. https://doi.org/10.3758/s13428-016-0809-y

References - suggested readings in bold (2)

Perugini, M., Gallucci, M., & Costantini, G. (2018). A Practical Primer To Power Analysis for Simple Experimental Designs. International Review of Social Psychology, 31(1). https://doi.org/10.5334/irsp.181

Preacher, K. J., & Hayes, A. F. (2004). SPSS and SAS procedures for estimating indirect effects in simple mediation models. Behavior Research Methods, Instruments, & Computers, 36(4), 717-731. https://doi.org/10.3758/BF03206553

Qiu, W. (2017). powerMediation: Power/sample size calculation for mediation analysis. R package version 0.2.9. Retrieved from https://cran.r-project.org/package=powerMediation

Schoemann, A. M., Boulton, A. J., & Short, S. D. (2017). Determining Power and Sample Size for Simple and Complex Mediation Models. Social Psychological and Personality Science, 8(4), 379-386. https://doi.org/10.1177/1948550617715068

Sobel, M. E. (1982). Asymptotic confidence intervals for indirect effects in structural equation models. Sociol Methodology, 13(1982), 290-312. https://doi.org/10.2307/270723

Zhang, Z., & Wang, L. (2013). bmem: Mediation analysis with missing data using bootstrap. R package version 1.5. Retrieved from https://cran.r-project.org/package=bmem