Standard Operating Procedures For Using Mixed-Effects Models

A Principled Workflow from the Decision, Development, and Psychopathology (D2P2) Lab
document version 1.0.0 -- 28 June 2020

[This document will be continuously updated and expanded; it may contain typos and other errors--both unintentional errors and errors based on incorrect or outdated knowledge--we will try to improve these things in future versions. Feel free to let us know if you spotted such things, how to further improve this document!]

Authors (in alphabetical order except that the youngsters were so kind to put the oldest guy in the lab first; BF)

Bernd Figner, Johannes Algermissen, Floor Burghoorn, Leslie Held, Afreen Khalid, Felix Klaassen, Farnaz Mosannenzadeh, Julian Quandt

Content/Analysis Steps

1. Before data collection: Power/ design/ planning/ sample size

Before starting data collection, whenever possible, we make use of power analysis, sensitivity analysis, and/or employ sequential sampling or other stopping rules to achieve an adequate sample size. A set of different approaches and tools for sample size calculation and planning the design is introduced below. Whatever approach from the list below is chosen, we strongly recommend taking this part seriously and pre-registering the sampling plan. In our lab, we do not all agree on one specific method. That's why we don't have a single recommendation for one method.

1.1. Power analysis

There are several approaches and tools for power analysis in mixed-effects models (some tools are similar to software like G*Power). Here, we group them into two general approaches.

The first approach is creating your own **simulation-based power analysis**. This approach is flexible and recommended for situations where more control over the parameter space (e.g. about the sampling errors) is wanted or when the other tools/apps (explained below) are not sufficient. There are several noteworthy resources regarding this approach, including:

- A very short step-by-step guide by Ben Bolker, best suited for people already familiar with data simulation.
- Another brief tutorial for custom simulations by Tood Jobe
- A tutorial paper by Tom Snijders
- simr**,** an R-package for calculating power for generalised linear mixed models, using simulation.
- simstudy, an R package for simulation-based power analysis (or, more generally, for simulating data) which can handle also more complex and clustered data (e.g., patients in therapists, in clinics, etc; possible to introduce different types of missingness etc); see also helpful example blog post here on how to best do pre/post comparisons with treatment and control group: https://www.rdatagen.net/post/thinking-about-the-run-of-the-mill-pre-post-analysis/
- longpower, an R-package for power-simulation of longitudinal data.
- Optimal design, a software to find the optimal research design.
- MLPowSim, an extensively annotated software for power simulation of mixed models. (https://sites.google.com/site/optimaldesignsoftware/home)

Another approach which is less flexible but perhaps easier to use is applying **power analysis tools/apps** such as:

- Power Analysis with crossed random effects by Jake Westfall for a design where, e.g., subjects and items are both random factors.
- Power Analysis with two random factors (crossed or nested) by Jake Westfall for similar purposes as the previous one, but more flexible.
- PANGEA, a comprehensive App by Jake Westfall for mixed ANOVA designs, in which within and/or between subject factors are present and factors can be nested in multiple levels.

- Simulating for LMEM, an app by Lisa DeBruine for power quasi-simulations in which each parameter of the model can be adjusted using a slider (allows for random factors for subjects and items).
    - R code by Lisa DeBruine for flexible data simulation
    - Tutorial paper by LisaDeBruine and Dale Barr for flexible data simulation (including logistic mixed-effects regression)
    - Blog post by Julian Quandt

Lastly, for direct replications, Murayama, Usami, and Sakaki (preprint) have argued to just use the t-test of the respective effect of a previous study and compute power as for a one-sample t-test.

1.2. Sensitivity analysis

In some contexts, it might be useful to use sensitivity analysis rather than power analysis. Sensitivity analysis takes a given sample size (and other relevant information such as number of trials, number of stimuli in the case of random effects for stimuli, etc.) as input and computes which effect sizes could be detected with the given sample (in contrast, conventional power analysis takes expected effect sizes as input and computes the required sample size). Recently, some journals, e.g., the Journal of Experimental Social Psychology, have adopted the editorial guideline to ask for sensitivity power analyses as a more objective alternative to the rather subjective choice of expected effect size estimates. Furthermore, sensitivity analysis appears to be more realistic in projects with limited budgets and/or time constraints, e.g., student projects. If a sensitivity analysis yields an effect size that seems reasonable, the study can be conducted; otherwise, this might be an indication that the research design and/or research question(s) need to be changed. In cases where the budget or time constraints are less strict, power analysis may be conducted with the tools previously described above.

1.3. Sequential sampling with stopping rules

Other approaches include flexible sampling plans that allow for sequential sampling until a stopping rule is met. These often take the precision of the parameter estimate of interest as a criterion for sufficient power (they can also be combined with pragmatic stopping rules such as time or budget constraints, again particularly relevant, e.g., for student projects).
Relevant references are:

- This blog post by Geoff Cumming
- This blog post by John Kruschke about optional stopping in a Bayesian context
- Stop doing sequential testing with Bayes factors by Corson N. Areshenkoff
- de Heide, R., & Grünwald, P. D. (2017). Why optional stopping is a problem for Bayesians. *arXiv preprint arXiv:1708.08278*. https://arxiv.org/abs/1708.08278
- Kelley, K., Darku, F. B., & Chattopadhyay, B. (2018). Accuracy in parameter estimation for a general class of effect sizes: A sequential approach. *Psychological Methods*, *23*(2), 226. https://psycnet.apa.org/record/2017-15257-001
- Lakens, D. (2014). Performing high-powered studies efficiently with sequential analyses. *European Journal of Social Psychology*, *44*(7), 701-710. https://onlinelibrary.wiley.com/doi/full/10.1002/ejsp.2023
- Rouder, J. N. (2014). Optional stopping: No problem for Bayesians. *Psychonomic Bulletin & Review*, *21*(2), 301-308. https://link.springer.com/article/10.3758/s13423-014-0595-4

- Schönbrodt, F. D., Wagenmakers, E. J., Zehetleitner, M., & Perugini, M. (2017). Sequential hypothesis testing with Bayes factors: Efficiently testing mean differences. *Psychological methods*, *22*(2), 322. https://psycnet.apa.org/record/2015-56330-001
- Schönbrodt, F. D., & Wagenmakers, E. J. (2018). Bayes factor design analysis: Planning for compelling evidence. *Psychonomic Bulletin & Review*, *25*(1), 128-142. https://link.springer.com/article/10.3758/s13423-017-1230-y
- For a more critical view, see this blogpost by Richard Morey: https://medium.com/@richarddmorey/power-and-precision-47f644ddea5e

1.4. More reading suggestions

More literature on power analysis:

- Brysbaert, M., & Stevens, M. (2018). Power analysis and effect size in mixed effects models: A tutorial. *Journal of Cognition*, *1*(1). http://www.journalofcognition.org/articles/10.5334/joc.10/
- Correll, J., Mellinger, C., McClelland, G. H., & Judd, C. M. (2020). Avoid Cohen's 'small', 'medium', and 'large' for power analysis. *Trends in Cognitive Sciences*. https://www.sciencedirect.com/science/article/abs/pii/S1364661319302979
- Lane, S. P., & Hennes, E. P. (2018). Power struggles: Estimating sample size for multilevel relationships research. *Journal of Social and Personal Relationships*, *35*(1), 7-31. http://journals.sagepub.com/doi/10.1177/0265407517710342

For more information about boosting power by increasing the number of trials (under certain conditions), see:

- Baker, D. H., Vilidaite, G., Lygo, F. A., Smith, A. K., Flack, T. R., Gouws, A. D., & Andrews, T. J. (2019). Power contours: Optimising sample size and precision in experimental psychology and human neuroscience. *arXiv preprint arXiv:1902.06122*. https://arxiv.org/abs/1902.06122
  - Associated App: https://shiny.york.ac.uk/powercontours/
- Boudewyn, M. A., Luck, S. J., Farrens, J. L., & Kappenman, E. S. (2018). How many trials does it take to get a significant ERP effect? It depends. *Psychophysiology*, *55*(6), e13049. http://doi.wiley.com/10.1111/psyp.13049
- Rouder, J. N., & Haaf, J. M. (2018). Power, dominance, and constraint: A note on the appeal of different design traditions. *Advances in Methods and Practices in Psychological Science*, *1*(1), 19-26. https://journals.sagepub.com/doi/full/10.1177/2515245917745058

For an extensive paper on Bayesian design planning:

- Schad, D. J., Betancourt, M., & Vasishth, S. (2019). Toward a principled Bayesian workflow in cognitive science. *arXiv preprint arXiv:1904.12765*. http://arxiv.org/abs/1904.12765

2. Preparing data

2.1. Categorical variables

We most commonly use sum-to-zero coding for categorical predictors (via the `options(contrasts = c("contr.sum", "contr.poly"))` for factors. We use this coding scheme because we are typically interested in *main effects* and *main interactions* rather than *simple effects* or *simple interactions* (see also this blog post by Dale Barr). One option is also to use the command `mixed()` from the package **afex**, as it will automatically set all contrasts to sum-to-zero.

Reasons to deviate might include the use of *custom contrasts* to test specific hypotheses.

We usually will *follow-up* on significant effects involving factors with more than two levels by either restricting analyses to only two levels in the form of follow-up models (i.e. analyzing a subset of the data comprising only two levels of the given factor) or, alternatively, we use some post-hoc procedures, e.g., using the package **emmeans** (for more details on both, see the section on post-hocs and follow-ups below).

2.2. Continuous variables

As a default, we typically use z-standardization for the continuous predictors (to help with model estimation), unless there are specific reasons not to do so (e.g. if we want to interpret effects on the original scale; in these cases, we typically center the predictor(s)). Centering is essential to make interactions interpretable and avoid so-called nonessential multicollinearity (Dunlap and Kemery, 1987; Marquardt, 1980; also see this blog post by Philipp Masur).

3. Running the model

3.1. Model specification and random effects

As a general guideline, we strive to follow the approach of fitting maximal models (in the sense of Barr et al., 2013), i.e., including all random intercepts, slopes, and correlations justified by the experimental design/the data structure.

We are aware that there is a possible trade-off between Type 1 errors (fitting maximal models should avoid inflated Type 1 errors; Barr et al., 2013) and Type 2 errors (maximal models can reduce power if they are too complex given the data, see Matuschek et al., 2017). In our studies, we are often more concerned about not inflating Type 1 error than about inflating Type 2 error and thus maximal models appear to be an appropriate default strategy.

However, if in a specific study, we prefer a different trade-off, we will make this explicit in the pre-registration of that study. A possible scenario might be the following: We test for some effect and it is not significant. However, the model is potentially too complex. In this context, the burden is to try and show as convincingly as possible that the effect is indeed not significant. Therefore, one could remove the random slope for the fixed effect of interest and see whether one still obtains a non-significant result. If it is still non-significant, we would be quite convinced that this non-significance is not due to a loss in power caused by an overly complex random-effects structure of the maximal model.

For clustering variables (e.g., subjects, items), a minimum of 5 levels (e.g. 5 subjects) should be available (better more, e.g., > 30); otherwise, we add these clustering factors just as fixed effects.

For control/nuisance variables, we try to add random slopes where appropriate. However, if we have convergence issues, we may opt to not add them as random slopes in order to reduce model complexity (but in such a case we will refrain from interpreting the associated *p*-value; see Barr et al., 2013).

3.2. Addressing convergence warnings

3.2.1. Convergence warnings in R's lme4

In case of convergence warnings, we attempt the listed approaches, typically in the order in which they are listed (these steps are based mainly on recommendations by Ben Bolker/the lme4 team and Barr et al., 2013). For each step, we check whether it resolves the convergence issues.
Before going through the listed steps, some of us would always set the optimizer to bobyqa as default via `optimizer = c("bobyqa"))` (since it has been suggested that it might work better for the kind of data that we typically have in psychology) and/or switch off the calculation of the gradient and Hessian via `control = [g]lmerControl(calc.derivs = FALSE)`; these settings might already resolve the convergence issues.

1. We increase the number of iterations to the maximum.
2. We use the estimates from the previous (non-converged) fit as our new starting values.
3. We compare the estimates of different optimizers (e.g., using `allFit()`); if different optimizers give highly similar estimates (even if they give convergence warnings), the convergence warnings can be considered false positives.
4. We follow the steps suggested in Ben Bolker's blog post:
    a. Center independent (and dependent) variables instead of scaling; multiply the independent variables by 10 (or 100) to increase the variance
    b. Robustness check: Check whether certain random correlations are close to +/-1 and/or certain random slope variances are close to 0. If yes, remove those; afterwards check whether the estimates are still the same
    c. Double check gradient calculations: Check the (parallel) minimum of the absolute and relative gradients. If those gradients are > 0.001, gradient calculation is likely not a problem.

If the above steps don't work, we try model simplifications:

1. We drop random effects in the following order: random correlations, random slopes of covariates (where significance is of no interest), random intercepts ("0+" instead "1+") (following Barr et al., 2013). We never remove the random slopes of the variables of interest (i.e., the ones for which we want to conduct significance tests).
   Please note that removing random correlation terms can be tricky if random slopes are estimated for factors with 3 or more levels. In that case, it is probably easiest to use `afex::mixed()` with `expand_re = TRUE` (an alternative option is to create manually the relevant contrasts yourself and add them as predictors to your model, which allows you to suppress the random corrections using the double pipe symbol ||).
2. We try to run separate analyses: For example, one model to only test the fixed and random effect of A (with fixed effect of B present); then one model to only test the effect of B.
   If we really have to drop random slopes, we follow the next step:
3. We follow the PCA approach suggested by **rePsychLing** (see Bates et al., 2015) that is performing a PCA on the random effects and following the guidelines described in the paper.
   a. We use a likelihood ratio test to test whether the model fit becomes significantly worse. As we prefer a more conservative approach here (i.e., rather err on the side of keeping too many random effects; we prioritize avoiding inflated Type 2 errors for this kind of decision), we use larger alpha-level of .2 (Matuschek et al., 2017).
   b. Alternatively, we suggest an Information criterion approach to avoid using a *p* value for our inclusion/exclusion decision, but choose the best model based on *BIC* or *AIC*.

As a last resort, we use:

- Two-stage regression
  (also called summary statistics approach, e.g. Gelman, 2005):
  Estimate a separate linear/logistic regression per participant, extract the regression coefficients, perform a one-sample *t*-test (or a two-sample *t*-test if testing for group differences) to test whether a certain regression coefficient is significantly different from zero on a group level.
  - This approach constitutes a special case of mixed models with stronger assumptions, i.e. all participants are assumed to provide equally reliable estimates and none of them is an outlier. Also, no shrinkage to the group-level mean is applied in such a case. See e.g. this comparison of both approaches bz Eshin Jolly.
  - This approach is very common in fMRI analyses.
  - As a slightly more sophisticated variant of the same idea, a meta analysis approach can be used to conduct the test across the per-participant regression coefficients, for example using the **metafor** package. The advantage is that this approach also carries forward the uncertainty (i.e., standard errors) from the first level (akin to meta-analysis).
- Sandwich estimator
  See e.g. the Huber-White sandwich estimator provided by the **merDeriv** package using the **sandwich** package.

3.2.2. Or we choose a Bayesian approach

As an alternative to targeting convergence issues within **lme4**, we suggest fitting the same model with **brms** and comparing it to the **lme4** fit. We assume that both provide similar results when qualitative conclusions regarding significant/non-significant effects are the same. Similarity checks can als be done regarding the parameter estimates. Thus, brms could either be used to check and verify a lme4 model with convergence issues, or brms could of course also be used instead of lme4 (different lab members have different preferences about this, so we will not make one single recommendation).

In **brms,** we investigate the convergence of chains by *at least* checking the following:

1. Trace-plots of the **brms** chains: `plot(model)`
    a. Did the chains converge (no change of variance across time & chains look like fat caterpillars)?
    b. Are the posterior distributions of the parameters of interest approximately normal and unimodal?
2. Are the Rhats (also sometimes spelled R-hat) reported in `summary(model)` between 0.99 and 1.01?
3. Are tail and bulk n-eff ("ESS" in summary output) big enough (bulk n-eff should be bigger than 100 times the number of chains (warnings will be provided if they are very low)?
4. Are no other convergence warnings issued (e.g. exceeding maximum tree-depth, divergent transitions)? If there are, check the Stan Manual convergence guide.

A more extensive tutorial on model-checking by Rens van der Schoot, provides additional information on how these things can be checked and what else might be worth investigating. In case of influential observations, instead of removing them, changing the model family (blog post by Solomon Kurz) provides a more robust alternative (see above).

3.2.3. MixedModels in Julia

As another alternative to **lme4** or **brms**, we might consider using `MixedModels` in **Julia** (currently, we do not have much experience with them in our lab). For more information, see also these links:

- https://github.com/RePsychLing/MixedModels-lme4-bridge/blob/master/using_jellyme4.ipynb
- https://github.com/JuliaStats/MixedModels.jl/
- https://github.com/palday/JellyMe4.jl

3.3. Important notes/considerations

3.3.1. Families/ distributions

When using **lme4** and `glmer()`, there are different options to specify a family, such as the Gaussian default, inverse Gaussian, binomial, or Poisson distributions.

Beyond those options, **brms** provides us with more freedom in specifying different model families, which translate our assumptions about the data-generating process into a distribution for the response variable. Ideally, the model family should be specified in the pre-registration. However though, if during model evaluation, it appears that the prespecified family does not fit the response distribution, the family might be changed post-analysis to increase the model fit with the data (see

Deciding on a family below). In these cases, it is essential to report and justify the deviations from the pre-registration and to point out possible disagreements between the pre-registered and the improved model(s).

3.3.1.1. Deciding on a family

- One approach is to generate a histogram/density plot of your raw DV (per condition/group). This gives a first indication of what the population distribution may look like. If you have different conditions/groups, consider plotting your DV for each group/condition separately, because the total distribution might be a mixture of separate sub-distributions.
- If there are multiple candidate distributions that might be appropriate, but we are not sure which one to use, we normally fit the same model with the different distributions separately and select the one that shows the best fit to the data (i.e., lowest AIC/BIC/some other deviance measure). To confirm that the model is healthy, check the *model diagnostics* (e.g., normality of residuals, see model diagnostics).
- See also this shiny-app by Jonas Lindeløv for a demonstration of the various distributions in brms that can be used to model reaction times.

3.3.1.2. Some commonly used families per DV type

- Continuous
  - Gaussian (default). Examples: amount of money offered/returned, some psychophysiological measures, quasi-continuous rating-scales (i.e. with many > 10 levels), speeded reaction times (without long tail)
    Robust alternative: Student.
  - (Shifted) lognormal / ex-gaussian / skewed normal. Examples: for skewed data such as reaction times, skin conductance responses, quasi-continuous rating-scales
- Categorical / Ordinal/ Counts with defined maximum
  - Bi- or multinomial/ Bernoulli. Examples: binary choice (approach/avoid; LL/SS; risky/sure, ambiguous/unambiguous); multinomial choice (healthy, neutral, unhealthy foods).
  - Cumulative. Examples: for ordinal data such as height (low/medium/high), size (small/medium/large), attractiveness (unattractive/neutral/attractive), rating-scales with few levels.
- Counts without maximum
  - Poisson. Examples: number of books sold within a week
  - Negative binomial

See also the documentation of the `family` and `brmsfamily` functions. Based on more anecdotal evidence from our lab, beta-binomial distributions work well for data bound between 0 and a maximum (e.g., rating data). Particularly for rating data, see also this helpful post here (on using zero-inflated beta models):
https://vuorre.netlify.app/post/2019/02/18/analyze-analog-scale-ratings-with-zero-one-inflated-beta-models/#zoib-regression

3.3.2. Estimation method: ML versus REML versus Bayesian

This is how we decide which estimation method to use:

- Bayesian versus (RE)ML:
  We have differing preferences in our lab and thus the individual pre-registrations will describe which approach each project will use. Some pros and cons involve that Bayesian methods are more flexible (e.g., in the terms of available families) but can be more time-consuming. Also, **brms** can sometimes fit models that are difficult to fit without convergence issues in **lme4**.
- ML versus REML:
  The default in **lme4** is REML and we use it unless we have good reasons to use ML instead (e.g., if we intend to use likelihood ratio tests or to solve convergence issues by switching to ML). Since there is a debate about whether ML or REML is more advantageous, in the future, we might change our position.

3.3.3. Priors when using a Bayesian approach

In a Bayesian approach, it is necessary to specify priors. In general, there are two routes that one can follow concerning priors, default priors or custom priors. Which one is better might depend on the situation.

Default priors

brms provides default priors that are *weakly regularizing*, which means that they somewhat constrain the possible parameter space to rule out vastly implausible parameter values, but do not comprise much commitment about the specific parameter value that we would expect. Using default priors is generally safe to do and they will not provide you with wrong conclusions in most cases. Using them might be a good idea if there is *no* information about the parameter space.
Please be aware that you must **not** use the default priors if you want to compute Bayes factors; you need informative priors for that. (At least some of us are skeptical of Bayes factors and do not use them anyway).

Custom priors

If there is anything that one can *a-priori* say about the parameter space (which in most cases is possible and easier than it might appear), it is often a good idea to specify custom priors, which can be tested for their implications by performing prior predictive checks. Specifying custom priors is especially useful when a previous study already provided data (such as in direct replication studies), in which case the posterior of the previous study can be used as the prior of the new study. For fixed-effects, normally distributed priors are often a good choice, while for random-effects, priors with heavier tails (e.g. Cauchy or Student-*t* distributed) might be more appropriate.

In our lab, we have different opinions about the use of default versus custom priors. Therefore, we prefer not to commit generally to one or the other and will specify this in the individual study pre-registrations. As a general rule, if in doubt, we use weakly regularizing priors (e.g., the default priors in brms). If we use custom priors, we check whether the different prior specifications lead to different results by comparing them to weakly regularizing (default) priors.

4. Model Diagnostics

In terms of diagnostics, there are many things one could possibly do. Thus, nearly everything reported below is optional.

As a rule of thumb, we always, at minimum, look at the following plots: qq-plots, density plots of residuals, and predicted versus observed values.

Note that we always perform our diagnostics on the *model residuals*, not the raw data.

If there are statistical (numeric) versus visual ways to inspect the data, we usually prefer visualisation. For example, commonly used tests like Kolmogorov-Smirnov tests are not appropriate for large enough datasets, and small *p*-values in such tests might be misleading when testing assumptions.

We recommend to check diagnostics in the following order, since fixing the former ones will often also fix the latter ones (based on a suggestion by Ben Bolker):

1. Outliers and influential cases
2. Non-linearity
3. Homoscedasticity
4. Normality
5. Plot fitted vs. observed

For more details on how these are implemented in code, check the appendix.

For the very handy package **performance**, containing many automated plots for model diagnostics, see, e.g., this vignette.

5. Inferring significance (*p*-values, *CIs*, Bayesian)

5.1. Frequentist approach (ML/ REML)

When using a frequentist approach, we typically obtain Type-III *p*-values in one of the following ways (see also, e.g., Luke, 2017; but see also Barr et al., 2013 showing that likelihood ratio tests seem trustworthy). In the *pre-registration* of an individual project, we determine beforehand which method we are using. Since methods sometimes fail, it might make sense to pre-register a *decision tree*, e.g., “we plan to use method x to determine *p*-values; if that fails for technical reasons, then we use method y as fallback; etc.”. If we had to recommend one specific method, then some of us would recommend KR *F*-tests and some would recommend Satterthwaite *F*-tests.

- *F*-test with Kenward-Roger approximation for degrees of freedom:
  Run using either the `Anova()` function of the package **car** (Fox & Weisberg, 2019) or using the `mixed()` function of the package **afex** (Singmann, Bolker, Westfall, Aust, & Ben-Shachar, 2019) with option `method = "KR"` (if you use `afex::mixed()`, then adding the argument `test_intercept = TRUE` means `car::Anova` is used in the background; otherwise, it will use `lmerTest`). These functions in turn call the `KRmodcomp()` function of the package **pbkrtest** (Halekoh & Højsgaard, 2014).

- *F*-test with Satterthwaite approximation for degrees of freedom:
  Run using the `mixed()` function of the package **afex** with option `method = "S"`, which in turn calls the package **lmerTest** (Kuznetsova, Brockhoff, & Christensen, 2017).
- (Bootstrapped) Likelihood Ratio Tests:
  Run using the `mixed()` function of the package **afex** with option `method = "LRT"`. If bootstrapped with option `method = "PB"`, this calls the function `PBmodcomp()` of the package **pbkrtest**. Note that LRTs are the only available option (other than t-as-z and Wald chi-square tests; both of which we try to avoid) to directly obtain *p*-values for models fit with `glmer()`.
- 95% confidence intervals:
  CIs can be used by inspecting whether the interval includes 0 or not. These should be based preferably either on bootstrapping or profiling the likelihood (both available via **lme4**). If necessary, *CI*s can then turned into *p*-values (e.g., if a 95% *CI* does not include zero, this can be used to derive that the *p*-value is < .05)

Note: whenever possible, we do **not** use t-as-z approaches, nor Wald chi-square tests (as implemented, e.g., in the `Anova()` function of the package **car**).

5.2. Bayesian approach

When using a Bayesian approach, we use the function `brm()` from the **brms** package (Bürkner, 2017) which provides an interface to Stan (Carpenter et al., 2017). A Bayesian model does not work with *p*-values to base the statistical significance of predictors on. There are several ways to compute null hypothesis significance testing (NHST) in a Bayesian framework, including the following:

- Computing 95% posterior density intervals (either via **brms** default method based on quantiles or HDI intervals, available, e.g., via packages **sjstats**, **tidybayes**; **HDInterval, bayestestR**; see this vignette by Makowski et al.; please be aware that **emmeans** computes HDI CIs, see below; it probably makes sense to decide on one method to compute CIs a-priori and then use that same method throughout all the analyses of a study/project). As our decision rule, we check whether the CI includes 0.
- Computing a Bayesian "*p*-value" based on the proportion of posterior samples larger or smaller than 0. Please think about whether you want to compute a one-sided or two-sided test and accordingly use the appropriate proportion of samples fulfilling that criterion.
- Some more hands-on in **brms**: we can use the command `summary('model-name')` to get the 95% credible interval (CI) by default. More specifically: per predictor, we get a coefficient, its estimated error, and the lower and upper end of the 95% CI range. If the 95% CI does not include 0, we deem an effect "significant" (i.e., we get a probability distribution of true values for a specific parameter; and if the 95% range of that distribution does not include 0, we deem it likely "enough" that the true value does not include 0 and call the effect significant). If we are interested in estimating trend effects or doing one-tailed tests (or computing any other CIs), we can get the 90% (or any other) CI by specifying `summary('model-name', prob=.90)`.
- If using the package **emmeans**, for pairwise comparisons/simple effects of the model (e.g., to find out for an interaction which levels significantly differ; Lenth, 2019), we get as output 95% HPD (highest posterior density) intervals which work the same way: if the 95% HPD does not include 0, the pairwise comparison or simple effect is significant.

There are other ways to test significance or find support for a hypothesis (see, e.g., for a discussion of several approaches Makowski et al., 2019). These methods also include Bayes Factors. For different approaches of how to compute Bayes Factors for mixed models, see. e.g.. this tutorial by Jonas Lindeløv. However, we are currently not using Bayes Factors as a default method in our lab, as some of us are quite skeptical. For critical discussions, including many code examples, see:

- the above-mentioned tutorial by Jonas Lindeløv,
- a series of blog posts by Richard Morey, see especially Part 2;
- a series of blog posts by Uri Simonsohn: http://datacolada.org/78a, http://datacolada.org/78b, and http://datacolada.org/78c;
- Bayes factors are almost impossible to use in practice by Corson N. Areshenkoff
- Dance of the Bayes Factors by Daniel Lakens
- The absurdity of mapping p-values to Bayes factors by Stephen R. Martin
- An explanation of the default Cauchy prior width of r = .707 used in JASP and the **BayesFactor** package by Eric-Jan Wagenmakers
- Why psychologists should not change the way they analyze their data: The devil is in the default prior by Ulrich Schimmack
- Wagenmakers' default prior is inconsistent with the observed results in psychological research by Ulrich Schimmack

For more information on indices of effect existence and significance in the Bayesian framework, see Makowski et al. (2019).

6. Post-hocs, follow-ups, simple slopes

Sometimes, to better understand the result patterns, we further investigate main effects or interactions by running additional analyses. In general, we use one of two approaches for additional analyses, post-hoc tests or follow-up models (for some pros and cons of each, see end of this section).

6.1. Post-hoc tests

The post-hoc tests that we use typically depend on the type of our predictors:

- For a significant categorical predictor with > 2 levels, we use the command `emmeans()`
- For a significant interaction between a categorical and continuous predictor, we use the commands `emtrends()` and `contrast(emtrends(), "pairwise", by = NULL)`.
- For a significant interaction between two categorical predictors, we use the commands `contrast(emmeans(), ‘pairwise’)` and `contrast(contrast(emmeans(), ‘pairwise’), 'pairwise', by=NULL)`.

For more details and code specifics, see the appendix.
You can also specify yourself which contrasts you want to test/compare, see, e.g., this vignette on how to use **emmeans**.

Note that **emmeans** (Lenth, 2019) can be used for **`lme4::`**`glm()` / **`afex::`**`mixed()` outputs as well as for Bayesian models (**brms**). It returns estimated marginal means per simple effect and can compute contrasts between them: For Bayesian models, it uses 95% highest posterior density or HPD intervals, while for **lme4**-type models, it provides *p*-values, which can be adjusted for multiple comparisons or not (adjustments for multiple tests are currently not available for brms models; if we want to adjust for multiple tests in brms models, we implement our own adjustment). For FAQs of **emmeans**, see the respective vignette.

6.2. Follow-up models

Another way to further investigate main effects or interactions is to run separate follow-up models. For example, if we find an interaction between a factor with 2 levels and/or several covariates, one can run 2 models, one per factor level. However, if we have an interaction that includes a factor with more than 2 levels, it would be necessary to run models where the more-than-two levels are restricted to just two levels, which means that multiple models will be run. Whether we adopt such a strategy of follow-up models or rather a post-hoc approach will be determined in the individual study pre-registration.

6.3. General advice

- We only run the follow-up/post-hoc tests that are relevant. We find it often sufficient to interpret the pattern of the interaction based on figures showing the pattern, rather than running many possible additional tests. In our opinion and experience, the main model is typically the most important one for drawing conclusions.
- **emmeans** uses the model estimates for post-hoc tests, not the raw data. Therefore, we always check with raw data or other methods whether the results/conclusions from our post-hocs seem reasonable.
- Correction for multiple comparisons can be done automatically in **emmeans** for ***lme4*** and ***afex*** models. This statement is *not the case for **brms**! Thus, if adjustment for multiple tests is desired for **brms** post-hoc tests, we do this ourselves.*
- When fitting separate models for different DVs, some kind of correction for multi-comparisons is often warranted. In such a situation it is worth considering approaches that might mitigate inflated Type 1 errors by means other than adjusting *p*-values: Gelman, Hill, & Yajima, 2012 describe a solution where the identity of the DV (e.g., different items or subscales in a questionnaire; when using DVs on different scales, it is appropriate to *standardize* those first) are used as a *grouping variable*. The shrinkage applied to the levels of this grouping variable will automatically adjust for multiple comparisons while retaining higher power. Another option to consider are multivariate mixed-effects models, which are quite easy to run in brms (and very flexible in that they allow the combination of DVs from different distributions, and also allow different predictors for different DVs). It is worth looking at the respective brms vignette.

6.4. More considerations

6.4.1. Omnibus vs. targeted tests

Although this does depend on our research question, in general we're interested in specific effects, and thus we strive to run targeted tests and not just omnibus tests. That being said, this might be different for different projects/research questions and thus the individual project's pre-registration will specify the testing strategy.

6.4.2. Contrasts

In general, it is often possible to modify the contrast coding (using custom contrasts) in such a way that the model directly tests the desired comparisons. This could make post-hoc and follow-up tests obsolete. For a nice treatment and tutorial, see Schad, Vasishth, Hohenstein, and Kliegl (2020).
For a tutorial of how to compute contrasts with brms, see this blog post by Matti Vuorre.

7. Reporting results

7.1. In Writing

Our reports include a description of the following parts (also see Meteyard & Davies, 2019; Barr et al., 2013):

- Model specification, including:
  - Dependent variable, and all fixed and random effects (intercepts, slopes, correlations), both in words and possibly also by providing the model equation/ R-pseudo code (so-called Wilkinson notation)
  - Transformation of variables, e.g., standardizing or centering variables
  - Contrast coding (typically sum-to-zero coding)
- Inference:
  - Description of how *p*-values were obtained (in case of a frequentist approach) or what other (Bayesian) decision rule was used for inference.
  - Description of what post-hoc or follow-up tests were performed
  - Any convergence issues that may arise while running the model (in particular if they require adjustments in the model specification) and how they were dealt with should be described, as well as the subsequent adjustments that were made.
- Model output, at minimum the following:
  - Model results: (un)standardized regression coefficients, standard errors and/or confidence / credible intervals, test statistics, degrees of freedom, *p*-values

7.2. Plotting

One question when plotting is how to compute the correct standard errors from the raw data (as there seems to be no generally accepted solution for all cases). One can thus either decide to plot

the model-based results, or decide to plot the raw data (these two do not always give the same impressions, and it might not be possible to compute appropriate standard errors/CIs for plotting the raw data).
Here are some options:

- For plotting regression coefficients (several of us find this a most informative plot, because it allows for comparisons across magnitudes and uncertainties of the different observed effects):
    - Use SEs/CIs from the model output.
- For plotting group/condition means:
    - If plotting the raw data (single data points), do not plot any indicator of uncertainty (i.e., no CI or SE indicator), unless there is an appropriate way to calculate it.
    - If aggregating raw data per condition, compute the SEs of the mean like in an ANOVA.
    - When plotting the raw data for within-subjects SEs, mind that between-subjects variability could/should be subtracted first and an appropriate correction for the potential bias performed (Morey, 2008). This is already implemented in the `summarySEwithin()` command from package **Rmisc** (see e.g. this blog post by Niklas Johannes, this blog post by Matt Craddock on visualizing ERPs, and an associated discussion on an MNE Python github issue). Please be aware that this is not a universally accepted approach.
    - Use model-based plots instead of plotting the raw data (e.g., the **effects** package; Fox, 2003).

7.3. A note on effect sizes

There are no generally accepted ways to compute standardized effect sizes for mixed effects models, but different variants have been proposed (such as Pseudo-R2; variants of Cohen's d, etc). Individual pre-registrations will specify if they want to report standardized effect sizes, and if so, which (and how they compute them).

References

Barr, D. J. (2013). Random effects structure for testing interactions in linear mixed-effects models. *Frontiers in Psychology, 4*, 328. https://www.frontiersin.org/articles/10.3389/fpsyg.2013.00328/full

Barr, D. J., Levy, R., Scheepers, C., & Tily, H. J. (2013). Random effects structure for confirmatory hypothesis testing: Keep it maximal. *Journal of Memory and Language*, 68(3), 255-278. https://www.sciencedirect.com/science/article/pii/S0749596X12001180

Bates, D., Kliegl, R., Vasishth, S., & Baayen, H. (2015). Parsimonious mixed models. *arXiv preprint arXiv:1506.04967.* https://arxiv.org/abs/1506.04967

Bürkner, P. C. (2017). Advanced Bayesian multilevel modeling with the R package brms. *arXiv preprint arXiv:1705.11123*. https://arxiv.org/abs/1705.11123

Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., ... & Riddell, A. (2017). Stan: A probabilistic programming language. *Journal of Statistical Software, 76*(1). https://www.jstatsoft.org/article/view/v076i01

Dunlap W. P., & Kemery E. R. (1987). Failure to detect moderating effects: is multicollinearity the problem? *Psychological Bulletin,* 102, 418–420. https://psycnet.apa.org/record/1988-06430-001

Fox, J. (2003). Effect Displays in R for Generalised Linear Models. *Journal of Statistical Software*, 8(15), 1–27. http://www.jstatsoft.org/v08/i15/

Fox, J., & Weisberg, S. (2019). *An R companion to applied regression (*3rd ed.) Thousand Oaks, CA: Sage. https://socialsciences.mcmaster.ca/jfox/Books/Companion/index.html

Gelman, A. (2005). Two-stage regression and multilevel modeling: a commentary. *Political Analysis*, *13*(4), 459-461. https://www.cambridge.org/core/journals/political-analysis/article/twostage-regression-and-multilevel-modeling-a-commentary/169E482D48466654556439FEC9EA6EA0

Gelman, A., Hill, J., & Yajima, M. (2012). Why we (usually) don't have to worry about multiple comparisons. *Journal of Research on Educational Effectiveness*, *5*(2), 189-211. https://www.tandfonline.com/doi/full/10.1080/19345747.2011.618213

Halekoh, U., & Højsgaard, S. (2014). A Kenward-Roger approximation and parametric bootstrap methods for tests in linear mixed models – The R Package pbkrtest. [Computer software]. *Journal of Statistical Software, 59,* 1-30. http://www.jstatsoft.org/v59/i09/

Kuznetsova, A., Brockhoff, P. B., & Christensen, R. H. B. (2017). lmerTest package: Tests in linear mixed effects models. [Computer software]. *Journal of Statistical Software, 82,* 1-26. doi: 10.18637/jss.v082.i13 https://www.jstatsoft.org/article/view/v082i13

Lenth, R. (2019). Emmeans package: Estimated marginal means, aka least-squares means. R package version 1.3. 5.1. https://cran.r-project.org/web/packages/emmeans/emmeans.pdf

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

Makowski, D., Ben-Shachar, M. S., Chen, S. H., & Lüdecke, D. (2019). Indices of effect existence and significance in the Bayesian framework. *Frontiers in Psychology*, *10*, 2767. https://www.frontiersin.org/articles/10.3389/fpsyg.2019.02767/full

Makowski, D., Ben-Shachar, M. S., & Lüdecke, D. (2019). *bayestestR: Describing Effects and their Uncertainty, Existence and Significance within the Bayesian Framework*. Journal of Open Source Software, 4(40), 1541. https://joss.theoj.org/papers/10.21105/joss.01541

Marquardt, D. W. (1980). Comment: You should standardize the predictor variables in your regression models. *Journal of the American Statistical Association, 75*(369), 87-91. https://amstat.tandfonline.com/doi/abs/10.1080/01621459.1980.10477430

Matuschek, H., Kliegl, R., Vasishth, S., Baayen, H., & Bates, D. (2017). Balancing Type I error and power in linear mixed models. *Journal of Memory and Language*, 94, 305-315. https://www.sciencedirect.com/science/article/pii/S0749596X17300013

Meteyard, L., & Davies, R. (2020). Best practice guidance for linear mixed-effects models in psychological science. *Journal of Memory and Language,* 112, https://doi.org/10.1016/j.jml.2020.104092

Morey, R. D. (2008). Confidence intervals from normalized data: A correction to Cousineau (2005). *Tutorial in Quantitative Methods for Psychology*, 2008, Vol. 4(2), p. 61-64. http://www.tqmp.org/RegularArticles/vol04-2/p061/p061.pdf

Singmann, H., Bolker, B., Westfall, J., Aust, F., & Ben-Schachar, M. S. (2020). afex: Analysis of factorial experiments. [Computer software]. https://CRAN.R-project.org/package=afex

Schad, D. J., Vasishth, S., Hohenstein, S., & Kliegl, R. (2020). How to capitalize on a priori contrasts in linear (mixed) models: A tutorial. *Journal of Memory and Language*, 110, 104038. https://www.sciencedirect.com/science/article/pii/S0749596X19300695

Wang, T., & Merkle, E. C. (2018). merDeriv: Derivative computations for linear mixed effects models with application to robust standard errors. *Journal of Statistical Software, Code Snippets*, *87*(1), 1-16. https://www.jstatsoft.org/article/view/v087c01

Appendix

Diagnostics

Outliers

- We save the standardized residuals
  ```
  sum(abs(resid(model, scaled = TRUE)) > value) /
  length(resid(model))
  ```

- We generally expect the following pattern (based on a normal distribution):
  o No values larger than +/- 3 (or 3.5)
  o Max. 1 % larger than +/- 2.5
  o Max. 5 % larger than +/- 2

Auto-correlation

- Use the function `acf():` We expect no significant lags (no bars more extreme than the dotted horizontal lines)
  - `library(lme4)`
  - `plot(acf(sleepstudy$Reaction)) # pretty dramatic autocorrelation in the raw data`

    - `m1 <- lmer(Reaction ~ Days + (1 + Days | Subject), data = sleepstudy)`
    - `plot(acf(resid(m1))) # no serious autocorrelation in the residuals`

Homoscedasticity

- Plot of fitted values vs. residuals to check for homo/heteroskedasticity (optional: fitted vs. observed values)
    - `plot(model, type = c('p', 'smooth'))`
- Check the ratio between the highest and lowest variance (by visual inspection, called Fmax).
- For ungrouped data (i.e. continuous predictors), heteroscedasticity is not fatal: “The linear relationship between variables is captured by the analysis, but there is even more predictability if the heteroscedasticity is accounted for. If it is not, the analysis is weakened, but not invalidated” (Tabachnick & Fidell, 2013, p. 85).
- For group data (i.e. factors), for equal cell sizes (up to a ratio of 1:4), an Fmax of up to 10 is acceptable (Tabachnick & Fidell, 2013, p. 86). If cell sizes are very uneven (say 1:9) and variance larger in smaller cells than bigger cells, Fmax as small as 3 can be associated with increased Type 1 error (Milligan, Wong, & Thompson, 1987)

Normality

- Density plot or qq-plots of residuals to check for normal distribution:
    - `densityplot(resid(model, scaled = TRUE))`
    - `qqmath(model, scaled = TRUE)`
    - `qqPlot(resid(model))`

More formal criteria for influential cases

We like to use the function: `lme4::influence` (package **dharma** for generalized models) to get influence statistics for formal inspection:

- `inf_model <- influence(model, "grouping factor")`
- `str(inf_model)`

To check for problematic values

- Cook’s distance: `cooks.distance(inf_model)`
    - values larger than 1
    - values larger than 4/N (grouping units)
    - Points that stand out
        - `plot(inf_model, which = 'cook', sort=T)`

- Dfbeta: `dfbetas(inf_model)`
    - Values larger than 1
    - Values larger than 2/sqrt(N)
    - Points that stand out
        - `plot(inf_model, which = 'dfbetas')`

Additional quantitative and visual checks

- Check distributions of raw data and residuals per cell (factor levels):
  - `with(dataframe, densityplot(~y | factor))`
  - `with(dataframe, densityplot(~ res_model | factor))`
- Create xy plots for regressors separately over groups:
  - `xyplot(res_model ~ regressor, data = dataframe, type = c('p', 'r', 'smooth'))`
- Screen groups separately:
  - `xyplot(y ~ regressor | grouping factor, data = df, type = c('p', 'r'))`
  - `xyplot(res_model ~ regressor | grouping factor, data = dataframe, type = c('p', 'r'))`

Bayesian

Outliers

- We save the standardized residuals
  - `resid_data <- data.frame(residuals(model, method = “posterior_predict”))`
  - `z_resids <- scale(resid_data$Estimate, scale = TRUE)`
  - `sum(abs(z_resids) > value) / length(z_resids)`

- We generally expect the following pattern (based on a normal distribution):

  o No values larger than +/- 3 (or 3.5)
  o Max. 1 % larger than +/- 2.5
  o Max. 5 % larger than +/- 2

Auto-correlation

- Use the function `acf()`: We expect no significant lags (no bars more extreme than the dotted horizontal lines)
  - `acf(z_resids)# see above for how to get these`

Homoscedasticity

- Plot of fitted values vs. residuals to check for homo/heteroskedasticity (optional: fitted vs. observed values)
  - `fitted_data <- data.frame(fitted(model))`
  - `z_fitted <- scale(fitted_data$Estimate, scale = TRUE)`
  - `pd <- data.frame(z_fitted = z_fitted, z_resids = z_resids) # see above for getting z_resids`
  - `ggplot(pd, aes(z_fitted, z_resids)) + geom_point() + geom_smooth(method = “loess”, se = FALSE)`
- Check the ratio between the highest and lowest variance (by visual inspection, called Fmax).

- For ungrouped data (i.e. continuous predictors), heteroscedasticity is not fatal: “The linear relationship between variables is captured by the analysis, but there is even more predictability if the heteroscedasticity is accounted for. If it is not, the analysis is weakened, but not invalidated” (Tabachnick & Fidell, 2013, p. 85).
- For group data (i.e. factors), for equal cell sizes (up to a ratio of 1:4), an Fmax of up to 10 is acceptable (Tabachnick & Fidell, 2013, p. 86). If cell sizes are very uneven (say 1:9) and variance larger in smaller cells than bigger cells, Fmax as small as 3 can be associated with increased Type 1 error (Milligan, Wong, & Thompson, 1987)

Normality

- Density plot or qq-plots of residuals to check for normal distribution:

  -
    ```
    densityplot(z_resids)
    ```

  -
    ```
    ggplot(pd, aes(sample = z_resids)) +
         geom_qq() +
         geom_qq_line()
         # see above for how to get pd and z_resids
    ```

Note: Diagnostics can also be done with posterior-predictive checks (which some of us like to use)

More formal criteria for influential cases

- We use the function: `loo::loo` (package **loo**) to get influence statistics for formal inspection.
- we start with:

  -
    ```
    loo_model <- loo(model)
    ```

  -
    ```
    print(loo_model)
    Computed from 16000 by 1758 log-likelihood matrix

             Estimate    SE
    elpd_loo  -6917.9 115.4
    p_loo       135.6  20.0
    looic     13835.8 230.8
    ------
    Monte Carlo SE of elpd_loo is NA.

    Pareto k diagnostic values:
                             Count Pct.    Min. n_eff
    (-Inf, 0.5]   (good)     1745  99.3%   1053
     (0.5, 0.7]   (ok)          8   0.5%   617
       (0.7, 1]   (bad)         4   0.2%   17
       (1, Inf)   (very bad)    1   0.1%   10
    See help('pareto-k-diagnostic') for details.
    ```
- In the above, we see that there are 5 bad and very bad (i.e. influential) observations. If there are only a few of these (less than 10), we can test their influence directly by refitting the model once for each observation using. This can take a lot of time if the model-fitting takes a long time:

  - `loo_new <- loo(model, reloo = TRUE, reloo_extra_args = list(cores = n_cores, chains = n_chains)`

- Again we will get a table like the one above. If the resulting `Monte Carlo SE of elpd_loo` is small compared to the other SEs in the table, the influence of these observations is not too strong.
- If we have too many influential observations (more than 10), loo will tell you that approximate loo might not work well anymore and k-fold cross validation should be used instead.
- Alternatively, if we want to check robustness of our results without however many influential cases, we can exclude all of them at once the following way (if d is the data that was used during model-fitting)
  - `influential_cases <- pareto_k_ids(loo_model, threshold = .7)`
  - `d_new <- d[-influential_cases, ]`
  - `model_new <- update(model, newdata = d_new)`

  Now we can see whether conclusions stay the same

Post-hoc tests

- For significant categorical predictor with >2 levels, we use the command `emmeans(model-name, pairwise ~ factor_with_e.g.3levels):`
  - Returns estimated marginal means (EMMs) per factor level, the pairwise comparisons between the 3 factor levels (e.g. level 1-2, level 1-3, and level 2-3), returning estimates, and lower/upper end of 95% HPD intervals.
  - To get 90% HPD intervals, we use the command `confint(emmeans(model-name, pairwise ~ factor_with_3levels), level = .90).`
  - If using a response transformation, results are on the transformed scale as well. But if responses are on the log or logit scale (e.g., such as when using binary dependent variables), we can ‘back-transform’ them to the original scale using the command `emmeans(model-name, pairwise ~ factor_with_e.g.3levels, type=’response’)`. However, note that it is not always the best approach (for more information, see this emmeans vignette).
- For a significant interaction between a categorical and continuous predictor, we use the following commands.
  - `emtrends(model-name, ~ factor_with_xlevels, var = 'continuous_predictor')`. It returns simple slopes of the continuous predictor per factor level, and their significance (e.g., is the continuous predictor significant per factor level)
  - `contrast(emtrends(model-name, ~ factor_with_xlevels, var = 'continuous_predictor'), "pairwise", by = NULL)`. It returns pairwise comparisons between the factor levels for the continuous predictor effect (e.g., do the slopes differ significantly between the factor levels, comparing slope 1-2, slope 1-3, etc.)

- For a significant interaction between two categorical predictors, we can use the following commands.
    - `contrast(emmeans(model-name, ~  factor1 | factor2), ‘pairwise’)`. It returns per level of factor 2 the significance of factor 1 (e.g., is the effect of factor 1 significant for each separate level of factor 2)
    - `contrast(contrast(emmeans(model-name, ~ factor1 | factor2), ‘pairwise’), 'pairwise', by=NULL)`. It returns pairwise comparisons between the factor 2 levels for factor 1 (e.g., is the effect of factor 1 significantly different between the factor 2 levels)

End of Appendix