Mixed Models in R - A Practical Introduction

Henrik Singmann

May 2019

Overview: Statistical Models in R

1. Identify probability distribution of data (more correct: of conditional distribution of the response)
   - pass appropriate distribution via `family` argument for GLMs/GLMMs
2. Make sure variables are of correct type via `str()`
3. Set appropriate contrasts (orthogonal contrasts if model includes interaction): `afex::set_sum_contrasts()`
4. Describe statistical model using `formula`
   - For mixed models: Identify maximal random-effects structure justified by design.
5. Fit model: pass `formula` and `data.frame` to corresponding modeling function (e.g., `lm()`, `glm()`)
6. Check model fit (e.g., inspect residuals) and model adequacy (i.e., for GLMs/GLMMs)
   - If mixed model shows singular fit, iteratively reduce random-effects structure.
   - For GLMs/GLMMs: R package `DHARMa`: https://cran.r-project.org/package=DHARMa
7. Test terms (i.e., main effects and interactions): Pass fitted model to `car::Anova()`
8. Follow-up tests:
   - Estimated marginal means: Pass fitted model to `emmeans::emmeans()`
   - Specify specific contrasts on estimated marginal means (e.g., `contrast()`, `pairs()`)
   - Tests of simple effects via `test()` or after splitting data.

- `afex` combines correct coding (3.), fitting (5.) and testing (7.):
  - ANOVAs: `afex::aov_car()`, `afex::aov_ez()`, or `afex::aov_4()`
  - (Generalized) linear mixed-effects models: `afex::mixed()`

`R` Formula Interface for Statistical Models: `~`

- `R` `formula` interface allows symbolic specification of statistical models, e.g. linear models: `lm(y ~ x, data)`
- Dependent variable(s) left of `~` (can be multivariate or missing), independent variables right of `~`:

Formula	Interpretation
`~ x` or `~1+x`	Intercept and main effect of `x`
`~ x-1` or `~0 + x`	Only main effect of `x` and no intercept (questionable)
`~ x+y`	Main effects of `x` and `y`
`~ x:y`	Interaction between `x` and `y` (and no main effect)
`~ x*y` or `~ x+y+x:y`	Main effects and interaction between `x` and `y`

- **Formulas behave differently for coninuous and categorical covariates!!**
  - Always use `str(data)` before fitting: `int` & `num` is continuous, `Factor` or `character` is categorical.
  - Categorical/nominal variables have to be `factor`s. Create via `factor()`.
- Categorical variables are transformed into numerical variables using contrast functions (via `model.matrix()`; see Cohen et al., 2002)
  - **If models include interactions, orthogonal contrasts (e.g., `contr.sum`) in which the intercept corresponds to the (unweighted) grand mean should be used**: `afex::set_sum_contrasts()`
  - Dummy/treatment contrasts (`R` default) lead to simple effects for lower order effects.
  - For linear models: Coding only affects interpretation of parameters/tests not overall model fit.
- For models with only numerical covariates, suppressing intercept works as expected.

- For models with categorical covariates, suppressing intercept or other lower-order effects often leads to very surprising results (and should generally be avoided).
- For models involving factors with more than two levels, **avoid inspecting parameter estimates** (i.e., avoid `summary()` or `coef()`).
  - Parameter estimates are only meaningful for factors with two levels.
  - Use `car::Anova()` or `afex` for estimating model and testing terms
  - Use `emmeans` for inspection of results on factor-levels and design cells

Tests of Model Terms/Effects with `car::Anova()`

- `car::Anova(model, type = 3)` general solution for testing effects.
- Type II and III tests equivalent for balanced designs (i.e., equal group sizes) and highest-order effect.
- Type III tests require orthogonal contrasts (e.g.,`contr.sum`); recommended:
  - For experimental designs in which imbalance is completely random and not structural,
  - Complete cross-over interactions (i.e., main effects in presence of interaction) possible.
- Type II are more appropriate if imbalance is structural (i.e., observational data).

Follow-Up Tests

- Make many plots to understand interactions (e.g., using `afex_plot()`)
- Choice of follow-up test after significant interactions based on research questions
  - Simple effects (e.g., main effect of one factor conditional on other factor[s])
  - Comparison of specific cell means
- Two approaches for follow-up tests:
  - Model based using `emmeans` (assumes assumptions hold and uses shared error term)
  - Splitting data and running separate models for each split (assumes assumptions do not hold, use separate error terms)
- When splitting data or using `emmeans::test()`, adjustment for multiple testing needs to be done by hand; e.g., pass p-values to `p.adjust()`

Follow-up Tests with `emmeans` (Formerly `lsmeans`)

- `emmeans(model, c("factor"))` (or `emmeans(model, ~factor)`) produces estimates marginal means (or least-square means for linear regression) for model terms (e.g., `emmeans(m6, c("education", "gender"))`).
- Additional functions allow specifying contrasts/follow-up tests on the means, e.g.:
  - `pairs()` tests all pairwise comparisons among means.
  - `contrast()` allows to define arbitrary contrasts on marginal means.
  - `test(..., joint = TRUE)` for joint tests (e.g., simple effects if using `by`).
  - For more examples see vignettes: https://cran.r-project.org/package=emmeans

ANOVAs with afex

- `afex` ANOVA functions require column with participant ID:
  - `afex::aov_car()` allows specification of ANOVA using `aov`-like formula. Specification of participant id in `Error()` term. For example:
    `aov_car(dv ~ between_factor + Error(id/within_factor), data)`
  - `afex::aov_4()` allows specification of ANOVA using `lme4`-like formula. Specification of participant id in random term. For example:
    `aov_4(dv ~ between_factor + (within_factor|id), data)`

  - `afex::aov_ez()` allows specification of ANOVA using characters. For example:
    `aov_ez("id", "dv", data, between = "between_factor", within = "within_factor")`
- All `afex` ANOVA functions return same results (only differ in how to specify)

Repeated-Measures, IID Assumption, & Pooling

- Ordinary linear regression, between-subjects ANOVA, and basically all standard statistical models share one assumption: Data points are *independent and identically distributed* (*iid*).
  - Independence assumption refers to residuals: After taking structure of model (i.e., parameters) into account, probability of a data point having a specific value is independent of all other data points.
  - Identical distribution: All observations sampled from same distribution.
- For repeated-measures independence assumption often violated, which can have dramatic consequences on significance tests from model (e.g., increased or decreased Type I errors).
- Three ways to deal with repeated-measures:
  1. *Complete pooling*: Ignore dependency in data (often not appropriate, results likely biased)
  2. *No pooling*: Two step procedure. 1. Separate data based on factor producing dependency and calculate separate statistical model for each subset. 2. Analysis of distribution of estimates from step 1. (Prone to overfitting which decreases precision of parameter estimates, estimation error accumulates in step 2, combination and analysis of individual estimates can be non-trivial if interest is in more than 1 parameter)
  3. *Partial pooling*: Analyse data jointly while taking dependency into account (gold standard, e.g., mixed models)

Mixed Models

- Mixed models extend regular regression models via *random-effects parameters* that account for dependencies among related data points.
- **Fixed Effects**
  - Overall or *population-level average* effect of specific model term (i.e., main effect, interaction, parameter) on dependent variable
  - Independent of stochastic variability controlled for by random effects
  - Hypothesis tests on fixed effect interpreted as hypothesis tests for terms in standard ANOVA or regression model
  - Possible to test specific hypotheses among factor levels (e.g., planned contrasts)
  - *Fixed-effects parameters*: Overall effect of specific model term on dependent variable
- **Random Effects**
  - *Random-effects grouping factors*: Categorical variables that capture random or stochastic variability (e.g., participants, items, groups, or other hierarchical-structures).
  - In experimental settings, random-effects grouping factors often part of design one wants to generalize over.
  - Random-effects factor out idiosyncrasies of sample, thereby providing a more general estimate of the fixed effects of interest.
  - *Random-effects parameters*:
    * Provide each level of random-effects grouping factor with idiosyncratic parameter set.
    * zero-centered offsets/displacements for each level of random-effects grouping factor
    * added to specific fixed-effects parameter
    * assumed to follow normal distribution which provides *hierarchical shrinkage*, thereby avoids over-fitting
    * should be added to each parameter that varies within the levels of a random-effects grouping factor (i.e., factor is *crossed* with random-effects grouping factor)
    * Note: random-effects parameters (i.e., random-slopes) can only be added to a parameter if there exist multiple data points (i.e., replications) for each level of random-effects grouping

factor and the parameter (e.g., each cell of corresponding factor or design-cell)

Random-Effects Parameters in `lme4/afex`

Formula	Interpretation
`(1|s)`	random intercepts for `s` (i.e., by-`s` random intercepts)
`(1|s) + (1|i)`	by-`s` and by-`i` (i.e., crossed) random intercepts
`(a|s)` or `(1+a|s)`	by-`s` random intercepts and by-`s` random slopes for `a` plus their correlation
`(a*b|s)`	by-`s` random intercepts and by-`s` random slopes for `a`, `b`, and the `a:b` interaction plus correlations among the by-`s` random effects parameters
`(0+a|s)`	by-`s` random slopes for `a` and no random intercept
`(a||s)`	by-`s` random intercepts and by-`s` random slopes for `a`, but no correlation (expands to: `(0+a|s) + (1|s)`)

Note. Suppressing the correlation parameters via `||` works only for numerical covariates in `lmer` and not for factors. `afex` provides the functionality to suppress the correlation also among factors if argument `expand_re = TRUE` in the call to `mixed()` (see also function `lmer_alt()`).

Examples:

```
mixed(dv ~ within_s_factor * within_i_factor + (within_s_factor|s) + (within_i_factor|i),
data, method = "S")
mixed(dv ~ within_s_factor + (within_s_factor||s), data, method = "S", expand_re = TRUE)
```

Crossed Versus Nested Factors

- Factor `A` is **crossed** with factor `B` if multiple levels of `A` appear within multiple levels of `B`. Note that this definition allows for missing values (i.e., it does not need to hold that all levels of `A` appear in all levels of `B`). For example:
  - Levels `a1`, `a2`, ... of `A` appear in `b1` of `B` and in `b2` of `B`, etc.
  - A within-subject factor (e.g., `congruency`) is crossed with the `participant` factor.
  - If each participant responds to a random subset of items and each item is responded to by several participants, `participant` and `item` are crossed.
- Factor `A` is **nested** within factor `B` if some levels of `A` appear only within specific levels of factor `B`. E.g.,:
  - Levels `a1`, `a2`, and `a3` of `A` appear only in `b1` of `B` and `a4`, `a5`, and `a6` of `A` appear only in `b2` of `B`
  - Participants are nested in a between-subjects factor (e.g., `group`), because each level of `participant` only provides data for one level of the factor.
  - If student can be member of one class only and several classes were observed, factor `student` is nested within factor `class`.
- Both dependency structures dealt with in same conceptual manner, via independent random effects-parameters. Specifically, both need independent random effects terms in model formula. For example:
  - For `students` nested within `class`, where each student has unique label (i.e., `student` id 1 is assigned to exactly one student and not to different students in different classes), at least:
    `... + (1|student) + (1|class)`
  - If additional factor `A` is crossed with `class`, but not with `student` (e.g., some students in each class receive treatment `a1`, some others `a2`), by-class random slopes need to be added:
    `... + (1|student) + (A|class)`

Hypothesis-Tests for Mixed Models

- `lme4::lmer` does not include p-values.
- `afex::mixed` provides four different methods:
  1. Kenward-Roger (`method="KR"`, default): Provides best-protection against anti-conservative results, requires a lot of RAM for complicated random-effects structures.
  2. Satterthwaite (`method="S"`): Similar to KR, but requires less RAM.
  3. Parametric-bootstrap (`method="PB"`): Simulation-based, can take a lot of time (can be speed-up using parallel computation).
  4. Likelihood-ratio tests (`method="LRT"`): Provides worst control for anti-conservative results. Can be used if all else fails or if all random-effects grouping factors have many levels (e.g., over 50).
- `afex::mixed` uses orthogonal contrasts per default. Necessary for categorical variables in interactions.

Random-Effects Structure

- Omitting random-effects parameters for model terms which vary within the levels of a random-effects grouping factor and for which random variability exists leads to non-iid residuals (i.e., ϵ) and anti-conservative results (e.g., Barr, Levy, Scheepers, & Tily, 2013).
- Safeguard is *maximal model justified by the design*.
- If maximal model is overparameterized, contains degenerate estimates, and/or singular fits, power of maximal model may be reduced and a reduced model may be considered (Bates et al., 2015; Matuschek et al., 2017); however, reducing model introduces unknown risk of anti-conservativity, and should be done with caution.
- Steps for running a mixed model analysis:
  1. Identify desired fixed-effects structure
  2. Identify random-effects grouping factors
  3. Identify *maximal model justified by the design*:
     - Which factors/terms vary within levels of (i.e. are crossed with) each random-effects grouping factor?
     - Are there replicates within factor levels (or parameters/coefficients) for levels of random-effects grouping factor?
  4. Choose method for calculating p-values and fit maximal model
  5. Iteratively reduce random-effects structure until all degenerate/zero-variance random-effects parameters are removed.
- If the maximal model shows critical convergence warnings, reducing random-effects structure probably indicated, even though this introduces unknown risk of anti-conservativity:
  - Start by removing the correlation among random-effects parameters
  - Remove random-effects parameters for highest-order effects with lowest variance
  - It can sometimes help to try different optimizers
  - Compare p-values/fixed-effects estimates across models (p-values from degenerate/minimal models are not reliable)

GLMMs: Mixed-models with Alternative Distributional Assumptions

- Not all data can be reasonable described by a normal distribution.
- Generalized-linear mixed models (GLMMs; e.g., Jaeger, 2008) allow for other distributions. For example:
  - Binomial distribution: Repeated-measures logistic regression
  - Poisson distribution for count data
  - Gamma distribution for non-negative data (e.g., RTs)
- GLMMs require specification of the conditional distribution of the response (`family`) and link function.
- Link function determines how values on untransformed scale are mapped onto response scale.
- Specification of random-effects structure conceptually identical as for LMMs.

- GLMMs only allow two methods for hypothesis testing: `"LRT"` or `"PB"`.
- Inspection of residuals/model fit more important for GLMMs than for LMMs: R package `DHARMa`
- Fit with `lme4::glmer` or `afex::mixed`, both require `family` argument (e.g., `family = binomial`): `mixed(prop ~ a * b + (a|s) + (b|i), data, weights = data$n, family = binomial, method = "LRT")` (Note: `data$n * data$prop` must produce integers; number of successes.)