A6. Lineare gemischte Modelle

Feste und zufällige Effekte, REML und Inferenz in der Praxis

Dr. Paul Schmidt

Um alle in diesem Kapitel verwendeten Pakete zu installieren und zu laden, kann man folgenden Code ausführen:

```
for (pkg in c("lme4", "lmerTest", "nlme", "glmmTMB", "emmeans",
              "broom.mixed", "agridat", "tidyverse")) {
  if (!require(pkg, character.only = TRUE)) install.packages(pkg)
}

library(lme4)
library(lmerTest)
library(nlme)
library(glmmTMB)
library(emmeans)
library(broom.mixed)
library(agridat)
library(tidyverse)
```

Lineare gemischte Modelle (Linear Mixed Models, LMMs) erweitern das gewöhnliche lineare Modell um einen zweiten Effekttyp: zufällige Effekte. Sie sind unverzichtbar, sobald die Daten gruppierte, geclusterte, genestete oder anderweitig nicht unabhängige Beobachtungen enthalten - was die überwiegende Mehrheit der geplanten Versuche in der Landwirtschaft und den Life Sciences abdeckt. Dieses Kapitel erklärt die zugrunde liegenden Konzepte und zeigt, wie man gemischte Modelle in R mit `lme4`, `lmerTest` und verwandten Paketen fittet. Mehrere der Design-Kapitel in dieser Sektion bauen auf diesen Ideen auf: Incomplete-Block-Designs wie das Alpha-Design und das Augmented-Design fitten beide zufällige Blockeffekte, und das Row-Column-Design verwendet zufällige Zeilen- und Spalteneffekte.

Die Kurzversion

Man fittet eine ANOVA oder Regression, und irgendetwas am Versuchsdesign sagt einem, dass nicht alle Beobachtungen wirklich unabhängig sind. Pflanzen stammen aus demselben Block, Messungen stammen vom selben Tier, Parzellen teilen sich eine Zeile im Feld. Wann immer ein Faktor eine solche Gruppierungsstruktur beschreibt und man nicht primär an den konkreten Stufen interessiert ist, die zufällig in den Daten vorkommen, ist dieser Faktor ein natürlicher Kandidat für einen zufälligen Effekt. Das Modell heißt dann lineares gemischtes Modell (Mixed Model), weil es feste und zufällige Effekte mischt.

Das am weitesten verbreitete R-Paket dafür ist `lme4`. Die Syntax sieht fast aus wie `lm()`, mit einem zusätzlichen Term in Klammern, der den zufälligen Effekt beschreibt:

```
mod <- lmer(Reaction ~ Days + (1 | Subject), data = sleepstudy)
summary(mod)
```

```
Linear mixed model fit by REML. t-tests use Satterthwaite's method [
lmerModLmerTest]
Formula: Reaction ~ Days + (1 | Subject)
   Data: sleepstudy
```

```
REML criterion at convergence: 1786.5

Scaled residuals:
    Min      1Q  Median      3Q     Max
-3.2257 -0.5529  0.0109  0.5188  4.2506

Random effects:
 Groups   Name        Variance Std.Dev.
 Subject  (Intercept) 1378.2   37.12
 Residual              960.5   30.99
Number of obs: 180, groups:  Subject, 18

Fixed effects:
            Estimate Std. Error       df t value Pr(>|t|)
(Intercept) 251.4051     9.7467  22.8102   25.79   <2e-16 ***
Days         10.4673     0.8042 161.0000   13.02   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Correlation of Fixed Effects:
     (Intr)
Days -0.371
```

Wir verwenden den integrierten `sleepstudy`-Datensatz aus `lme4`, der Reaktionszeiten von 18 Probanden über 10 aufeinanderfolgende Tage Schlafentzug enthält. Der feste Teil `Reaction ~ Days` beschreibt den durchschnittlichen Effekt des Schlafentzugs auf die Reaktionszeit. Der Term `(1 | Subject)` fügt pro Proband einen zufälligen Achsenabschnitt hinzu: Jeder der 18 Probanden darf seine eigene Grundreaktionszeit haben, und diese Grundwerte werden als Ziehungen aus einer Normalverteilung modelliert.

> **Schnelle Entscheidungsregel**
>
> Wenn ein Faktor die Randomisierungsstruktur des Versuchs beschreibt (Blöcke, Parzellen, Tiere, Messzeitpunkte), behandelt man ihn als zufällig. Wenn ein Faktor eine Behandlung beschreibt, deren konkrete Stufen man vergleichen oder im Bericht benennen möchte, behandelt man ihn als fest.

Was ist ein zufälliger Effekt?

In jedem bisher betrachteten linearen Modell ist der Fehlerterm `e` selbst eine Zufallsvariable. Streng genommen enthält fast jedes Modell daher eine stochastische Komponente. Was einen zufälligen Effekt vom Fehler unterscheidet, ist, dass der zufällige Effekt die Variation zwischen Gruppen von Beobachtungen strukturiert, nicht zwischen einzelnen Beobachtungen.

Ein nützliches Gedankenmodell ist die Frage, woher die Stufen eines Faktors kommen. Bei einem festen Faktor sind die Stufen in den Daten genau die Stufen, die einen interessieren. Wenn der Behandlungsfaktor drei Sorten A, B, C hat, dann sind A, B, C die gesamte Geschichte. Bei einem zufälligen Faktor sind die Stufen in den Daten eine Stichprobe aus einer größeren Population möglicher Stufen. Die 18 Probanden in `sleepstudy` sind nicht 18 konkret ausgewählte Menschen, deren Reaktionszeiten uns interessieren - sie stehen stellvertretend für die Population potenzieller Probanden. Blöcke in einem Feldversuch sind ein weiteres klassisches Beispiel: Der Versuch hätte genauso gut eine andere Anordnung

von Blöcken verwenden können, und die Inferenz soll über die konkret verwendeten Blöcke hinaus verallgemeinern.

Eine häufige Frage ist, wie viele Stufen ein Faktor braucht, bevor er sinnvoll als zufällig behandelt werden kann. Eine Faustregel in der Literatur setzt das Minimum irgendwo zwischen 5 und 12 Stufen an, weil die Varianzkomponente, die die Streuung der zufälligen Effekte beschreibt, bei sehr wenigen Stufen schlecht geschätzt wird. Diese Regel ist eine Orientierung, keine harte Schwelle. Zwei Einschränkungen sind in der Praxis wichtig:

- Faktoren, die die Designstruktur beschreiben (Blöcke, Parzellen, Probanden), werden fast immer als zufällig behandelt, unabhängig davon, wie viele Stufen sie haben, weil die wissenschaftliche Interpretation es erfordert. Ein RCBD mit nur drei Blöcken als zufällig zu fitten ist gängige Praxis.
- Der Bereich von 5 bis 12 ist eine Konvention, kein Theorem. Worauf es tatsächlich ankommt, ist die Stabilität der Varianzschätzung, die von Stichprobengröße, Balance und der Größe der Varianzkomponente selbst abhängt.

H. P. Piepho, A. Büchse, and K. Emrich [1] geben eine ausführliche Diskussion über Entscheidungen zwischen fest und zufällig im Kontext der Pflanzenzüchtung, wo derselbe Faktor (Genotyp) je nach wissenschaftlicher Fragestellung sinnvoll auf beide Arten modelliert werden kann.

Allgemeine Darstellung

Ein klassisches lineares Modell lässt sich kompakt schreiben als

$$y = X\beta + e$$

wobei `y` der Vektor der Beobachtungen ist, `X` die Designmatrix der festen Effekte, `beta` der Vektor der Koeffizienten der festen Effekte und `e` der Vektor der Residuen. Ein lineares gemischtes Modell fügt einen weiteren Term hinzu:

$$y = X\beta + Zu + e$$

Hier ist `Z` die Designmatrix für die zufälligen Effekte und `u` der Vektor der zufälligen Effekte. Die Verteilungsannahmen lauten, dass `u` aus einer multivariaten Normalverteilung mit Mittelwert null und einer Varianz-Kovarianz-Matrix `G` stammt und dass `e` multivariat normalverteilt ist mit Mittelwert null und Varianz-Kovarianz-Matrix `R`. Unter diesem Aufbau sind die Beobachtungen `y` selbst multivariat normalverteilt mit Mittelwert `X*beta` und einer kombinierten Varianz `V = Z*G*Z' + R`, die sowohl die Variation der zufälligen Effekte als auch die Residualvariation erfasst.

Die wichtige Konsequenz ist, dass Beobachtungen, die sich eine Stufe eines zufälligen Effekts teilen, korreliert sind. Zwei Reaktionszeiten desselben `sleepstudy`-Probanden sind ähnlicher als zwei Reaktionszeiten verschiedener Probanden. Das gemischte Modell berücksichtigt diese Korrelation automatisch, anstatt so zu tun, als existiere sie nicht.

Ein kleines numerisches Beispiel

Ein winziger Datensatz hilft zu sehen, wie die Designmatrizen tatsächlich aussehen. Man betrachte drei Sorten mit je zwei Beobachtungen:

```
toy <- data.frame(
  var = factor(c("A", "A", "B", "B", "C", "C")),
  block = factor(c(1, 2, 1, 2, 1, 2)),
  yield = c(3.2, 3.6, 2.8, 2.9, 4.1, 4.0)
)
toy
```

```
  var block yield
1   A     1   3.2
2   A     2   3.6
3   B     1   2.8
4   B     2   2.9
5   C     1   4.1
6   C     2   4.0
```

Ein Modell mit Sorte als fest und Block als zufällig kann wie folgt gefittet werden:

```
mod_toy <- lmer(yield ~ var + (1 | block), data = toy)
```

```
boundary (singular) fit: see help('isSingular')
```

```
summary(mod_toy)
```

```
Linear mixed model fit by REML. t-tests use Satterthwaite's method [
lmerModLmerTest]
Formula: yield ~ var + (1 | block)
   Data: toy

REML criterion at convergence: 0.1

Scaled residuals:
    Min      1Q  Median      3Q     Max
-1.1547 -0.2887  0.0000  0.2887  1.1547

Random effects:
 Groups   Name        Variance Std.Dev.
 block    (Intercept) 0.00     0.0000
 Residual             0.03     0.1732
Number of obs: 6, groups:  block, 2

Fixed effects:
            Estimate Std. Error      df t value Pr(>|t|)
(Intercept)   3.4000     0.1225  3.0000  27.761 0.000103 ***
varB         -0.5500     0.1732  3.0000  -3.175 0.050270 .
varC          0.6500     0.1732  3.0000   3.753 0.033053 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Correlation of Fixed Effects:
     (Intr) varB
varB -0.707
varC -0.707  0.500
optimizer (nloptwrap) convergence code: 0 (OK)
boundary (singular) fit: see help('isSingular')
```

Die Designmatrix der festen Effekte `X` hat sechs Zeilen (eine pro Beobachtung) und drei Spalten (Achsenabschnitt plus zwei Sorten-Kontraste). Die Designmatrix der zufälligen Effekte `Z` hat ebenfalls sechs Zeilen, aber zwei Spalten - eine pro Block - und besteht aus Nullen und Einsen, die angeben, welche Beobachtung zu welchem Block gehört. Die Ausgabe von `summary()` berichtet die Schätzungen der festen Effekte, die geschätzte Varianz der Blockeffekte und die Residualvarianz.

Fest oder zufällig: Wie man entscheidet

Die Entscheidung zwischen fest und zufällig ist gelegentlich philosophisch, aber in der angewandten Arbeit klären sie eine Handvoll Leitfragen meist schnell.

Man behandelt einen Faktor als **zufällig**, wenn mindestens eines der folgenden zutrifft:

- Der Faktor repräsentiert eine Stichprobe aus einer größeren Population, auf die man verallgemeinern möchte. Die Auswahl einiger weniger Standorte aus einer Zielregion oder ein Panel von Probanden aus einer Population potenzieller Probanden sind Lehrbuchfälle.
- Der Faktor repräsentiert eine Randomisierungseinheit im Versuchsdesign. Blöcke in einem RCBD, Großparzellen (Main Plots) in einem Split-Plot-Design und unvollständige Blöcke in einem Alpha-Design fallen alle unter diese Regel.
- Der Faktor repräsentiert Teilstichproben einer Versuchseinheit: mehrere Pflanzen pro Parzelle, mehrere Blätter pro Pflanze, Messwiederholungen am selben Tier.
- Der Faktor ist mit einem anderen zufälligen Faktor gekreuzt. Wenn `Location` zufällig ist, dann erbt die Interaktion `Year:Location` die zufällige Eigenschaft.
- Man möchte den Stufen eine bestimmte Kovarianzstruktur auferlegen, zum Beispiel genetische Verwandtschaft zwischen Genotypen oder räumliche Nähe zwischen Parzellen.

Man behandelt einen Faktor als **fest**, wenn die konkreten Stufen das wissenschaftliche Ziel der Analyse sind. Behandlungsstufen in einem kontrollierten Experiment qualifizieren sich fast immer dafür: Wenn es im Versuch darum geht, drei Düngemittel zu vergleichen, sind diese drei Düngemittel fest, selbst wenn sie aus einem Katalog mit Dutzenden von Produkten ausgewählt wurden.

Pakete für gemischte Modelle in R

Drei Pakete decken die überwältigende Mehrheit der Arbeit mit gemischten Modellen in R ab.

Das **`lme4`-Paket** [2] ist der De-facto-Standard für lineare und generalisierte lineare gemischte Modelle mit unabhängigen Residuen und einfachen Strukturen der zufälligen Effekte. Seine Hauptfunktion ist `lmer()` für Gaußsche Zielvariablen und `glmer()` für nicht-Gaußsche Zielvariablen. Es ist schnell, in der Praxis erprobt und gut dokumentiert.

Das **`nlme`-Paket** [3] ist älter, aber immer noch weit verbreitet. Seine Funktion `lme()` unterstützt von Haus aus Modelle mit korrelierten Residuen und heterogenen Varianzen (über die Argumente `correlation =` und `weights =`), was `lme4` nicht tut. Wenn man AR(1)-Residuen für Messwiederholungen, Compound Symmetry oder Varianzfunktionen wie `varIdent` oder `varPower` braucht, ist `nlme::lme()` oft der direkteste Weg.

Das **`glmmTMB`-Paket** kombiniert einen Großteil der Syntax von `lme4` mit der Flexibilität von `nlme`. Es bewältigt eine breite Palette von Verteilungen der Zielvariablen (zero-inflated, negativ-binomial, Beta, Tweedie), unterstützt korrelierte zufällige Effekte und Residuen und skaliert gut auf größere Datensätze. Es ist die natürliche Wahl, wenn `lme4` zu einschränkend ist, man aber die Formel-Syntax im `lme4`-Stil beibehalten möchte.

Für die Zwecke dieses Kapitels konzentrieren wir uns auf `lme4`, weil seine Formel-Syntax zur Lingua franca der Spezifikation gemischter Modelle in R geworden ist.

Syntax der zufälligen Effekte in `lme4`

Der Term in Klammern auf der rechten Seite der Formel spezifiziert die zufälligen Effekte. Einige gängige Muster:

- `(1 | group)` - zufälliger Achsenabschnitt pro Gruppe. Jede Stufe von `group` erhält ihren eigenen Grundwert, gezogen aus einer Normalverteilung.
- `(1 | group1) + (1 | group2)` - zwei separate (gekreuzte oder genestete) zufällige Achsenabschnitte. Subject und Item in psycholinguistischen Experimenten sind der kanonische Fall.
- `(1 | group1/group2)` - `group2` genestet in `group1`. Äquivalent zu `(1 | group1) + (1 | group1:group2)`.
- `(slope | group)` - zufälliger Achsenabschnitt und zufällige Steigung (Random Slope) von `slope` pro Stufe von `group`, mit einer geschätzten Korrelation zwischen beiden.
- `(0 + slope | group)` - zufällige Steigung ohne zufälligen Achsenabschnitt.

Die `sleepstudy`-Daten sind ein gutes Testfeld für zufällige Steigungen. Probanden unterscheiden sich vermutlich nicht nur in der Grundreaktionszeit, sondern auch darin, wie stark der Schlafentzug sie beeinflusst:

```
mod_slope <- lmer(Reaction ~ Days + (Days | Subject), data = sleepstudy)
summary(mod_slope)
```

```
Linear mixed model fit by REML. t-tests use Satterthwaite's method [
lmerModLmerTest]
Formula: Reaction ~ Days + (Days | Subject)
   Data: sleepstudy

REML criterion at convergence: 1743.6

Scaled residuals:
    Min      1Q  Median      3Q     Max
-3.9536 -0.4634  0.0231  0.4634  5.1793

Random effects:
 Groups   Name        Variance Std.Dev. Corr
 Subject  (Intercept) 612.10   24.741
          Days         35.07    5.922   0.07
 Residual             654.94   25.592
Number of obs: 180, groups:  Subject, 18

Fixed effects:
            Estimate Std. Error      df t value Pr(>|t|)
(Intercept)  251.405      6.825  17.000  36.838  < 2e-16 ***
Days          10.467      1.546  17.000   6.771 3.26e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Correlation of Fixed Effects:
     (Intr)
Days -0.138
```

Der Abschnitt zu den zufälligen Effekten in der Ausgabe zeigt nun eine Varianz für den Achsenabschnitt, eine Varianz für die `Days`-Steigung und eine Korrelation zwischen beiden. Die Schätzung des festen Effekts von `Days` ist die durchschnittliche Steigung über alle Probanden.

ML und REML

Gewöhnliche lineare Modelle schätzen ihre Parameter über die Methode der kleinsten Quadrate. Gemischte Modelle können das nicht: Die Anwesenheit zufälliger Effekte bedeutet, dass die Varianzkomponenten selbst aus den Daten geschätzt werden müssen, und die Methode der kleinsten Quadrate hat dazu nichts zu sagen. Der Standardansatz ist die **Maximum-Likelihood-Methode (ML)**, die Parameterwerte wählt, die die beobachteten Daten unter der angenommenen Verteilung so wahrscheinlich wie möglich machen. Für ein rein festes Gaußsches Modell liefern ML und die Methode der kleinsten Quadrate identische Schätzungen für `beta`.

ML hat einen unangenehmen Nebeneffekt für gemischte Modelle: Es unterschätzt Varianzkomponenten, oft erheblich in kleinen Stichproben. Die Intuition ist, dass ML nicht für die Freiheitsgrade korrigiert, die durch die Schätzung der festen Effekte verbraucht werden, sodass die übrig bleibende Varianz kleiner aussieht, als sie tatsächlich ist. Dies ist analog dazu, die Residuenquadratsumme durch `n` statt durch `n - p` zu teilen, wenn man die Residualvarianz in einem linearen Modell schätzt.

Die **Restricted-Maximum-Likelihood-Methode (REML)** behebt dieses Problem. REML schätzt die Varianzkomponenten aus einer Version der Likelihood, die von den festen Effekten bereinigt wurde, und liefert unverzerrte oder zumindest deutlich weniger verzerrte Varianzschätzungen. Der Preis dafür ist, dass REML-Log-Likelihoods von der Struktur der festen Effekte abhängen, was eine praktische Konsequenz hat: **Likelihood-Quotienten-Tests und Informationskriterien (AIC, BIC), die auf REML-Fits basieren, sind nur dann sinnvoll, wenn die Struktur der festen Effekte über die verglichenen Modelle hinweg identisch ist.**

Die praktische Regel lautet:

- Man verwendet **REML** (die Voreinstellung in `lmer()` und `lme()`) zum Berichten von Varianzkomponenten, Standardfehlern, Konfidenzintervallen und Vorhersagen.
- Man verwendet **ML**, wenn man Modelle mit unterschiedlichen Strukturen der festen Effekte über Likelihood-Quotienten-Tests oder AIC/BIC vergleicht.

Der Wechsel zwischen beiden ist in `lme4` ein einzelnes Argument:

```
mod_reml <- lmer(Reaction ~ Days + (1 | Subject), data = sleepstudy)           # REML (default)
mod_ml   <- lmer(Reaction ~ Days + (1 | Subject), data = sleepstudy, REML = FALSE)

c(REML = logLik(mod_reml), ML = logLik(mod_ml))
```

```
     REML        ML 
-893.2325 -897.0393 
```

BLUEs und BLUPs

Gemischte Modelle erzeugen zwei konzeptionell verschiedene Arten von Schätzungen. Schätzungen fester Effekte heißen **Best Linear Unbiased Estimators (BLUEs)**. Sie sind im klassischen Sinne unverzerrt: Der Erwartungswert des Schätzers entspricht dem wahren Parameterwert.

Vorhersagen zufälliger Effekte heißen **Best Linear Unbiased Predictors (BLUPs)**. Sie sagen die (unbeobachteten) realisierten Werte der zufälligen Effekte vorher. BLUPs sind nicht im selben Sinne unverzerrt wie BLUEs; sie sind gegenüber dem, was man erhielte, wenn man denselben Faktor als fest fittete, “geschrumpft” (shrinkage) in Richtung null. Das Shrinkage ist stärker, wenn die Daten auf Gruppenebene spärlich oder verrauscht sind, und schwächer, wenn die Gruppe gut repräsentiert ist. Dieses Borrowing-of-Strength über Gruppen hinweg ist einer der wesentlichen praktischen Vorteile gemischter Modelle, besonders in unbalancierten oder unvollständigen Designs wie Alpha-Designs und Augmented-Designs.

Die Funktion `ranef()` extrahiert BLUPs, und `fixef()` extrahiert BLUEs:

```
fixef(mod)               # BLUEs
```

```
(Intercept)        Days
  251.40510    10.46729
```

```
head(ranef(mod)$Subject) # BLUPs for the first few subjects
```

```
    (Intercept)
308   40.783710
309  -77.849554
310  -63.108567
330    4.406442
331   10.216189
332    8.221238
```

In der Pflanzenzüchtung ist der BLUE eines Genotyps (aus einem Modell mit festen Effekten) die richtige Größe für die Sortenregistrierung oder einen direkten Vergleich zweier Genotypen, während der BLUP (aus einem Modell mit zufälligen Effekten) die richtige Größe für die Selektion ist, weil das Shrinkage verrauschte Genotyp-Schätzungen korrekt herunterwichtet. Die Entscheidung zwischen fest und zufällig für Genotypen wird daher von der wissenschaftlichen Fragestellung diktiert, nicht von einer allgemeinen Präferenz.

Inferenz und p-Werte

Eine bekannte Eigenheit von `lme4` ist, dass `summary()` keine p-Werte für feste Effekte berichtet, und `anova()` ebenfalls nicht:

```
anova(mod)
```

```
Type III Analysis of Variance Table with Satterthwaite's method
     Sum Sq Mean Sq NumDF DenDF F value    Pr(>F)
Days 162703  162703     1   161   169.4 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Das ist kein Fehler. Douglas Bates, der Autor von `lme4`, hat die p-Werte absichtlich entfernt, weil es keine universell korrekte Methode gibt, die Freiheitsgrade für die t- und F-Statistiken in einem allgemeinen linearen gemischten Modell zu berechnen. Für balancierte, orthogonale Designs gelten die üblichen Lehrbuchformeln, aber für unbalancierte Daten, gekreuzte zufällige Effekte oder komplexe Varianzstrukturen sind die “richtigen” Nenner-Freiheitsgrade ein offenes Problem. Einen p-Wert mit den falschen Freiheitsgraden zu berichten kann ernsthaft irreführend sein, daher weigert sich `lme4` zu raten.

Es existieren mehrere vernünftige Approximationen, und das `lmerTest`-Paket fügt sie auf `lme4` aufbauend hinzu. Das Laden von `lmerTest` ändert das Verhalten von `lmer()` so, dass `summary()` und `anova()` beginnen, p-Werte zu berichten:

```
# lmerTest was loaded with the packages block above
mod_lt <- lmer(Reaction ~ Days + (1 | Subject), data = sleepstudy)
anova(mod_lt)
```

```
Type III Analysis of Variance Table with Satterthwaite's method
     Sum Sq Mean Sq NumDF DenDF F value    Pr(>F)
Days 162703  162703     1   161   169.4 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Standardmäßig verwendet `lmerTest` die **Satterthwaite-Approximation** für die Freiheitsgrade. Eine konservativere Alternative ist die **Kenward-Roger-Approximation**, die zusätzlich die Varianz-Kovarianz-Matrix der festen Effekte für die Verzerrung bei kleinen Stichproben anpasst:

```
anova(mod_lt, ddf = "Kenward-Roger")
```

```
Type III Analysis of Variance Table with Kenward-Roger's method
     Sum Sq Mean Sq NumDF DenDF F value    Pr(>F)
Days 162703  162703     1   161   169.4 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Kenward-Roger wird im Allgemeinen für kleine oder unbalancierte Datensätze bevorzugt, um den Preis einer langsameren Berechnung. Satterthwaite ist eine vernünftige Voreinstellung für größere Datensätze.

> ! Warum `anova(lmer_model)` ohne `lmerTest` keine p-Werte hat
>
> Ohne geladenes `lmerTest` berichtet `anova()` bei einem `lmerMod`-Objekt nur F-Statistiken und keine p-Werte, weil `lme4` keine Nenner-Freiheitsgrade liefert. Dies ist die oben beschriebene bewusste Designentscheidung, kein fehlendes Feature. Das Laden von `lmerTest` - oder die direkte Verwendung von `pbkrtest::KRmodcomp()` - ist der Standard-Workaround.

Für Vergleiche geschätzter marginaler Mittelwerte greift das `emmeans`-Paket sowohl auf `lme4` als auch auf `lmerTest` zu und berechnet p-Werte automatisch mit Satterthwaite- oder Kenward-Roger-Freiheitsgraden:

```
emmeans(mod_lt, ~ Days, at = list(Days = c(0, 5, 9)))
```

```
 Days emmean   SE   df lower.CL upper.CL
    0    251 9.75 22.8      231      272
    5    304 9.06 17.1      285      323
    9    346 9.75 22.8      325      366

Degrees-of-freedom method: kenward-roger
Confidence level used: 0.95
```

Ein realistisches Beispiel: Alpha-Design

Das abstrakte `sleepstudy`-Beispiel ist nützlich für die Syntax, aber gemischte Modelle glänzen bei geplanten Versuchen mit Blockbildung. Das `agridat`-Paket liefert den klassischen Alpha-Design-Datensatz von J. John and E. Williams [4] mit:

```
dat <- agridat::john.alpha
str(dat)
```

```
'data.frame':   72 obs. of  7 variables:
 $ plot : int  1 2 3 4 5 6 7 8 9 10 ...
 $ rep  : Factor w/ 3 levels "R1","R2","R3": 1 1 1 1 1 1 1 1 1 1 ...
 $ block: Factor w/ 6 levels "B1","B2","B3",..: 1 1 1 1 2 2 2 2 3 3 ...
 $ gen  : Factor w/ 24 levels "G01","G02","G03",..: 11 4 5 22 21 10 20 2 23 14 ...
 $ yield: num  4.12 4.45 5.88 4.58 4.65 ...
 $ row  : int  1 2 3 4 5 6 7 8 9 10 ...
 $ col  : int  1 1 1 1 1 1 1 1 1 1 ...
```

Das Design hat 24 Genotypen in 2 Wiederholungen, unterteilt in unvollständige Blöcke von je 4 Parzellen. Eine naive RCBD-Analyse würde die Incomplete-Block-Struktur ignorieren:

```
mod_rcbd <- lm(yield ~ gen + rep, data = dat)
anova(mod_rcbd)
```

```
Analysis of Variance Table

Response: yield
          Df  Sum Sq Mean Sq F value    Pr(>F)
gen       23 14.0765 0.61202  4.5475 6.259e-06 ***
rep        2  6.1355 3.06774 22.7939 1.322e-07 ***
Residuals 46  6.1910 0.13459
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Eine korrekte Alpha-Design-Analyse behandelt die unvollständigen Blöcke (genestet innerhalb der Wiederholung) als zufälligen Effekt, was Information aus den Vergleichen innerhalb der Blöcke zurückgewinnt und typischerweise die Präzision erhöht:

```
mod_alpha <- lmer(yield ~ gen + rep + (1 | rep:block), data = dat)
anova(mod_alpha)
```

```
Type III Analysis of Variance Table with Satterthwaite's method
     Sum Sq Mean Sq NumDF  DenDF F value    Pr(>F)
gen 10.6786 0.46429    23 34.736  5.4478 4.376e-06 ***
rep  1.5703 0.78513     2 10.394  9.2124  0.004992 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Die Varianzkomponente für `rep:block` quantifiziert, wie viel der Gesamtvariation durch die unvollständigen Blöcke erklärt wird:

```
VarCorr(mod_alpha) %>% as.data.frame()
```

```
        grp        var1 var2       vcov     sdcor
1 rep:block (Intercept) <NA> 0.06194388 0.2488853
2  Residual        <NA> <NA> 0.08522511 0.2919334
```

Für die vollständige Analyse einschließlich Genotyp-Mittelwerten und Buchstabendarstellungen (Letter Displays) siehe das eigene Alpha-Design-Kapitel.

Praktischer Workflow

Beim Fitten eines gemischten Modells für einen geplanten Versuch bewährt sich in der Praxis folgender Workflow:

1. Das Design aufschreiben: Was sind die Behandlungen, was sind die Blockfaktoren, was ist die Randomisierungseinheit?
2. Für jeden Faktor mit den obigen Kriterien entscheiden, ob fest oder zufällig. Behandlungen sind meist fest, Faktoren der Designstruktur meist zufällig.
3. Das Modell mit `lmer()` fitten (oder `lme()` / `glmmTMB()`, falls die Situation es erfordert). REML für Varianzkomponenten und BLUPs verwenden.
4. Das Modell mit Diagnoseplots prüfen. Es gelten dieselben Residuenplots aus A1. Modelldiagnostik, mit dem Zusatz, dass die `ranef()`-Werte ungefähr normalverteilt sein sollten.
5. `lmerTest` laden und `anova()` mit Kenward-Roger-Freiheitsgraden für kleine oder unbalancierte Daten ausführen, ansonsten mit Satterthwaite.
6. Behandlungsmittelwerte mit `emmeans` vergleichen und BLUEs berichten.
7. Varianzkomponenten neben der Inferenz der festen Effekte berichten, weil sie die Verallgemeinerbarkeit des Versuchs beschreiben.

Weiterführende Ressourcen

Allgemein

- `lme4` JSS-Paper: D. Bates, M. Mächler, B. Bolker, and S. Walker [2]
- `nlme`-Buch: J. C. Pinheiro and D. M. Bates [3]
- GLMM FAQ von Ben Bolker - eine regelmäßig aktualisierte Troubleshooting-Referenz
- Douglas Bates' R-help-Beitrag dazu, warum `lme4` keine p-Werte hat: nach "lmer, p-values and all that" suchen

Fest versus zufällig

- H. P. Piepho, A. Büchse, and K. Emrich [1] für eine züchtungsorientierte Diskussion

Inferenz

- Dokumentation des `lmerTest`-Pakets für Satterthwaite
- Dokumentation des `pbkrtest`-Pakets für Kenward-Roger und parametrischen Bootstrap

Erweiterungen

- `glmmTMB`-Vignetten für Zero-Inflation, nicht-Gaußsche Zielvariablen und korrelierte Residuen
- `emmeans`-Vignetten für marginale Mittelwerte und Kontrastfamilien

ℹ Wichtigste Erkenntnisse

1. **Gemischte Modelle fügen Modellen mit festen Effekten zufällige Effekte hinzu.** Zufällige Effekte beschreiben eine Gruppierungsstruktur und werden als Ziehungen aus einer Normalverteilung modelliert.
2. **Designgetriebene Faktoren** (Blöcke, Parzellen, Probanden) sind zufällig, unabhängig davon, wie viele Stufen sie haben. Die 5-bis-12-Regel ist eine Orientierung für stichprobenbasierte Faktoren, keine harte Schwelle.
3. `lme4::lmer()` ist das Standard-Arbeitstier. `nlme::lme()` wird bevorzugt, wenn korrelierte Residuen oder Varianzfunktionen benötigt werden. `glmmTMB` deckt alles Übrige ab.
4. **REML** zum Berichten von Varianzkomponenten und Vorhersagen; **ML** für Likelihood-Quotienten-Tests, die unterschiedliche Strukturen der festen Effekte vergleichen.
5. **BLUEs** sind Schätzungen fester Effekte (unverzerrt). **BLUPs** sind Vorhersagen zufälliger Effekte (in Richtung null geschrumpft, Borrowing of Strength über Gruppen hinweg).
6. `lme4` **lässt p-Werte bewusst weg.** `lmerTest` für Satterthwaite-Approximationen (schnell) oder Kenward-Roger-Approximationen (genauer in kleinen Stichproben) laden.
7. `anova()` **ohne** `lmerTest` **liefert keine p-Werte** - das ist Absicht, kein Fehler.

Bibliography

[1] H. P. Piepho, A. Büchse, and K. Emrich, “A Hitchhiker's Guide to Mixed Models for Randomized Experiments,” *Journal of Agronomy and Crop Science*, vol. 189, no. 5, pp. 310–322, 2003, doi: 10.1046/j.1439-037X.2003.00049.x.

[2] D. Bates, M. Mächler, B. Bolker, and S. Walker, “Fitting linear mixed-effects models using lme4,” *Journal of Statistical Software*, vol. 67, no. 1, pp. 1–48, 2015, doi: 10.18637/jss.v067.i01.

[3] J. C. Pinheiro and D. M. Bates, *Mixed-Effects Models in S and S-PLUS*. New York: Springer, 2000. doi: 10.1007/b98882.

[4] J. John and E. Williams, “Cyclic and Computer Generated Designs,” *Biometrical Journal*, vol. 38, no. 7, p. 778, 1995, doi: 10.1002/bimj.4710380703.