Inference in mixed models in R - beyond the usual asymptotic likelihood ratio test

Søren Højsgaard\textsuperscript{1}
Ulrich Halekoh\textsuperscript{2}

June 12, 2018

\textsuperscript{1}University of Aalborg, Denmark
\textsuperscript{2}University of Southern Denmark, Denmark
Outline and take-home message

- Mixed models (random effects, random regression etc.) models handled by \texttt{lme4} package in R.
- Tests are based on $\chi^2$ approximation of LR test statistic.
  - Works fine with “large samples” / “large dataset”
  - But a dataset can be large with respect to some aspect of a model while small with respect to other.
- Package \texttt{pbkrtest} provides some remedies:
  - Base test on F-statistic, where denominator degrees of freedom are estimated from data.
  - Base test of parametric bootstrap where data are simulated under the model.
  - Parametric bootstrap carries over to e.g. generalized linear mixed models.
- Look at simulated and real data
- Shortcomings of \texttt{pbkrtest}
History: The degree-of-freedom police

- Years ago, Ulrich Halekoh and SH colleagues at “Danish Institute for Agricultural Sciences”
  - Main concern: Help protect researcher colleagues from reporting effects to be “more significant than they really are”.
  - Many studies called for random effects models - and for PROC MIXED (from SAS)
  - PROC MIXED reports (by default) $p$–values from asymptotic likelihood ratio test.
  - Common advice: Account for uncertainty in estimate of variance by doing $F$-test instead. Use Satterthwaite or Kenward-Roger approximation of denominator degrees of freedom in $F$-test – in an attempt not to get things “too wrong”.

- Then R became popular;
  - Mixed models fitted with nlme and lme4 package
  - No Satterthwaite or Kenward-Roger approximation, so our common advice fell apart.
SH raised the issue on R-help - 2006: [R] how calculation degrees freedom see:

- SH: Along similar lines ... probably in recognition of the degree of freedom problem. It could be nice, however, if anova() produced ...
- Doug Bates: I don’t think the “degrees of freedom police” would find that to be a suitable compromise. :-)

In reply to related question:

- Doug Bates: I will defer to any of the “degrees of freedom police” who post to this list to give you an explanation of why there should be different degrees of freedom.

The point being:

- Quite different views on whether the degree-of-freedom issue really is an issue or not.
Example: Double registration in labs

Clustered data:

- Compare two groups (treatment with a control);
- \( M \) units (petri plates, persons, animals...) per group;
- Each unit is measured \( R \) times. Measurements on same unit are positively correlated.
Simulated data: $N = 3$ subjects per group, $R = 2$ replicated measurements per subject.

<table>
<thead>
<tr>
<th></th>
<th>y1</th>
<th>y2</th>
<th>grp</th>
<th>subj</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>1.70</td>
<td>0</td>
<td>ctrl</td>
<td>subj1</td>
</tr>
<tr>
<td>#2</td>
<td>2.01</td>
<td>0</td>
<td>ctrl</td>
<td>subj1</td>
</tr>
<tr>
<td>#3</td>
<td>0.65</td>
<td>0</td>
<td>ctrl</td>
<td>subj2</td>
</tr>
<tr>
<td>#4</td>
<td>1.39</td>
<td>2</td>
<td>ctrl</td>
<td>subj2</td>
</tr>
<tr>
<td>#5</td>
<td>0.31</td>
<td>1</td>
<td>ctrl</td>
<td>subj3</td>
</tr>
<tr>
<td>#6</td>
<td>0.94</td>
<td>0</td>
<td>ctrl</td>
<td>subj3</td>
</tr>
<tr>
<td>#7</td>
<td>0.55</td>
<td>0</td>
<td>trt1</td>
<td>subj4</td>
</tr>
<tr>
<td>#8</td>
<td>1.20</td>
<td>2</td>
<td>trt1</td>
<td>subj4</td>
</tr>
<tr>
<td>#9</td>
<td>4.49</td>
<td>4</td>
<td>trt1</td>
<td>subj5</td>
</tr>
<tr>
<td>#10</td>
<td>4.53</td>
<td>5</td>
<td>trt1</td>
<td>subj5</td>
</tr>
<tr>
<td>#11</td>
<td>3.94</td>
<td>2</td>
<td>trt1</td>
<td>subj6</td>
</tr>
<tr>
<td>#12</td>
<td>4.02</td>
<td>0</td>
<td>trt1</td>
<td>subj6</td>
</tr>
</tbody>
</table>
Problem/issues: If we ignore clustering/positive correlation:

▶ pretending to have more information than we have
▶ standard errors of estimates become too small
▶ $p$ values become too small
▶ effects appear stronger than they really are.

Notice:

▶ Measuring the same unit many many times will make the dataset larger, but will not really add many more chunks of information (depending on the size of the within-subject correlation, of course).
▶ Instead, more units are needed.
library(tidyverse)

lg1 <- lm(y1 ~ grp, data=dub)
lg1 %>% summary %>% coef %>% as.data.frame -> tb1

tb1$"Pr(>|X^2|)" = 1 - pchisq(tb1[,3]^2, df=1)

tb1

#>                Estimate Std. Error t value Pr(>|t|) Pr(>|X^2|)
#> (Intercept)     1.167     0.5437   2.146   0.05747   0.03189
#> grp.grp1        1.955     0.7689   2.543   0.02923   0.01100

Notice: the \( t \)-test “accounts for” the uncertainty in the estimate of the standard error.
Alternative: Analyze average

duba <- aggregate(y1 ~ grp + subj, FUN=mean, data=dub)
lm(y1 ~ grp, data=duba) %>% summary %>% coef

|            | Estimate | Std. Error | t value | Pr(>|t|) |
|------------|----------|------------|---------|----------|
| (Intercept)| 1.167    | 0.8416     | 1.386   | 0.2380   |
| grptrt1    | 1.955    | 1.1903     | 1.642   | 0.1758   |

- Works fine (gives the correct test) in (nearly) balanced cases.
- Does not provide estimate of between and within subject variation (not necessarily severe problem here).
- Analyzing-the-average is often not a feasible strategy.
Alternative: Random effects model

```r
lg2 <- lmer(y1 ~ grp + (1|subj), data=dub)
tidy(lg2)
```

```r
## Warning in bind_rows_(x, .id): binding factor and character vector, ## coercing into character vector
## Warning in bind_rows_(x, .id): binding character and factor vector, ## coercing into character vector
## # A tibble: 4 x 5
## term estimate std.error statistic group
## <chr>    <dbl>    <dbl>      <dbl> <chr>
## 1 (Intercept) 1.17       0.842     1.39 fixed
## 2 grp.trt1    1.96       1.19      1.64 fixed
## 3 sd_(Intercept).subj 1.44      NA       NA subj
## 4 sd_Observation.Residual 0.350     NA       NA Residual
```

```r
sm2 <- update(lg2, .~. -grp)
as.data.frame(anova(lg2, sm2))
```

```r
## Df   AIC   BIC logLik deviance Chisq Chi Df Pr(>Chisq)
## sm2  3 36.80 38.26  -15.40   30.80   NA   NA    NA
## lg2  4 35.71 37.65  -13.86   27.71  3.093  1     0.07864
```

Notice: Test is based in the $\chi^2$ distribution (i.e. that the variance is known)
Alternatives in the pbkrtest package:

KRmodcomp(lg2, sm2)

```r
## F-test with Kenward-Roger approximation; time: 0.13 sec
## large : y1 ~ grp + (1 | subj)
## small : y1 ~ (1 | subj)
## stat ndf ddf F.scaling p.value
## Ftest 2.7 1.0 4.0 1 0.18
```

PBmodcomp(lg2, sm2)

```r
## Bootstrap test; time: 7.19 sec; samples: 1000; extremes: 186;
## large : y1 ~ grp + (1 | subj)
## small : y1 ~ (1 | subj)
## stat df p.value
## LRT 2.9 1 0.089
## PBtest 2.9 0.187
```

Notice: Same \( p \)-value as when analyzing average.
The Kenward–Roger approach

The Kenward–Roger modification of the F–statistic

For multivariate normal data

\[ Y_{n \times 1} \sim N(X_{n \times p} \beta_{p \times 1}, \Sigma) \]

we consider the test of the hypothesis

\[ L_{d \times p}(\beta - \beta_0) = 0 \]

With \( \hat{\beta} \sim N_d(\beta, \Phi) \), a Wald statistic is

\[ W = [L(\hat{\beta} - \beta_0)]^T [L \Phi L^T]^{-1} [L(\hat{\beta} - \beta_0)] \]

which is asymptotically \( W \sim \chi^2_d \) under the null hypothesis.
A scaled version of $W$ is

$$F = \frac{1}{d} W$$

- Asymptotically $F \sim \frac{1}{d} \chi^2_d$ under the null hypothesis.
- Think of as the limiting distribution of an $F_{d,m}$-distribution as $m \to \infty$.
- To account for the fact that $\Phi = \text{Var}(\hat{\beta})$ is estimated from data, we must come up with a better estimate of the denominator degrees of freedom $m$ (better than $m = \infty$).
- That was what Kenward and Roger worked on...
The linear hypothesis $L \beta = \beta_0$ can be tested via the Wald-type statistic

$$F = \frac{1}{r} (\hat{\beta} - \beta_0)^\top L^\top (L^\top \Phi(\hat{\sigma})L)^{-1} L(\hat{\beta} - \beta_0)$$

- $\Phi(\sigma) = (X^\top \Sigma(\sigma)X)^{-1} \approx \text{Cov}(\hat{\beta})$, $\hat{\beta}$ REML estimate of $\beta$
- $\hat{\sigma}$: vector of REML estimates of the elements of $\Sigma = \text{Var}(Y)$

Kenward and Roger (1997)

- replaced $\Phi$ by an improved small sample approximation $\Phi_A$
- scaled $F$ by a factor $\lambda$
- determined denominator degrees of freedom $m$ by matching moments of $F/\lambda$ with an $F_{d,m}$ distribution.
Shortcomings of Kenward-Roger

- The Kenward–Roger approach is no panacea.
- In the computations of the degrees of freedom we need to compute
  
  \[ G_j \Sigma^{-1} G_j \]

  where \( \Sigma = \sum_i \sigma_i G_i \). Can be space and time consuming!
- An alternative is a Satterthwaite–kind approximation which is faster to compute. Will come out in next release of \texttt{pbkrtest} (code not tested yet). Way faster...
- What to do with generalized linear mixed models – or even with generalized linear models.
- \texttt{pbkrtest} also provides the parametric bootstrap \( p \)-value. Computationally somewhat demanding, but can be parallelized.
**Parametric bootstrap**

We have two competing models; a large model $f_1(y; \theta)$ and a null model $f_0(y; \theta_0)$; the null model is a submodel of the large model.

▶ The $p$ value for a composite hypothesis is

$$p = \sup_{\theta \in \Theta_0} Pr_\theta(T \geq t_{obs})$$

where the sup is taken under the hypothesis.

▶ We can (usually) not evaluate the sup in practice, so instead we do:

$$p^{PB} = Pr_{\hat{\theta}}(T \geq t_{obs})$$

▶ In practice we approximate $p^{PB}$ as

▶ Draw $B$ parametric bootstrap samples $t^1, \ldots, t^B$ under the fitted null model $\hat{\theta}_0$.
▶ Fit the large and the null model to each of these datasets;
▶ Calculate the LR-test statistic for each simulated data; this gives reference distribution.
▶ Calculate how extreme the observed statistic is.
lg2 <- update(lg2, REML=FALSE)
sm2 <- update(sm2, REML=FALSE)

# Observed test statistic:
t.obs <- 2 * (logLik(lg2) - logLik(sm2))
t.obs

## 'log Lik.' 3.093 (df=4)

# Reference distribution
set.seed(121315)
t.sim <- PBrefdist(lg2, sm2, nsim=2000)

# p-value
head(t.sim)

## [1] 0.35260 2.40216 0.02194 1.20877 1.04064 1.88052

sum(t.sim >= t.obs) / length(t.sim)

## [1] 0.1685

# compare with X ~ 2 dist
1 - pchisq(t.obs, df=1)

## 'log Lik.' 0.07864 (df=4)
Interesting to overlay limiting $\chi^2_1$ distribution and simulated reference distribution.

Bootstrap reference distribution has heavier tail giving larger $p$-value.
Speedup I: Sequential $p$-value

Instead of simulating a fixed number of values $t^1, \ldots, t^B$ for determining the reference distribution used for finding $p^{PB}$ we may instead introduce a stopping rule saying simulate until we have found, say $h = 20$ values $t^j$ larger than $t_{obs}$. If $J$ simulations are made then the reported $p$–value is $h/J$.

```r
spb <- seqPBmodcomp(lg2, sm2)
spb

## Bootstrap test; time: 1.53 sec;samples: 200; extremes: 34;  
## large : y1 ~ grp + (1 | subj)  
## small : y1 ~ (1 | subj)  
##      stat df p.value  
##    LRT   3.09  1 0.079  
##   PBtest 3.09   0.174
```
Parametric bootstrap is computationally demanding, but multiple cores can be exploited. Done by default on linux / mac platforms.

```
PBmodcomp(lg2, sm2)  # Default: Use all cores (4 on my computer)
```

```
## Bootstrap test; time: 9.86 sec; samples: 1000; extremes: 177;
## large : y1 ~ grp + (1 | subj)
## small : y1 ~ (1 | subj)
## stat df p.value
## LRT  3.09  1  0.079
## PBtest 3.09  0.178
```

```
PBmodcomp(lg2, sm2, cl=1)  # Use one core
```

```
## Bootstrap test; time: 15.13 sec; samples: 1000; extremes: 179;
## large : y1 ~ grp + (1 | subj)
## small : y1 ~ (1 | subj)
## stat df p.value
## LRT  3.09  1  0.079
## PBtest 3.09  0.180
```
On windows (in fact, work on all platforms):

```r
set.seed(121315)
library(parallel)
nc <- detectCores(); nc
clus <- makeCluster(rep("localhost", nc))
PBmodcomp(lg2, sm2, cl=clus)
```
Speedup III: Parametric form of reference distribution:

Estimating tail–probabilities will require more samples than estimating the mean (and variance) of the reference distribution.

Suggests to approximate simulated reference distribution with a known distribution so that fewer samples will suffice:

```r
pb1 <- PBmodcomp(lg2, sm2, nsim=1000)
pb2 <- PBmodcomp(lg2, sm2, nsim=100)
summary(pb1) %>% as.data.frame

## stat df ddf p.value
## LRT 3.093 1 NA 0.07864
## PBtest 3.093 NA NA 0.20480
## Gamma 3.093 NA NA 0.19419
## Bartlett 1.688 1 NA 0.19382
## F 3.093 1 4.404 0.14685

summary(pb2) %>% as.data.frame

## stat df ddf p.value
## LRT 3.093 1 NA 0.07864
## PBtest 3.093 NA NA 0.18812
## Gamma 3.093 NA NA 0.18422
## Bartlett 1.760 1 NA 0.18461
## F 3.093 1 4.641 0.14348
```
Why use parametric bootstrap

- Applies generally; in pbkrtest implemented for e.g. generalized linear mixed models (where random effects are on the linear predictor scale).
- Kenward-Roger does not readily scale to larger problems because of the computation of

\[ G_j \Sigma^{-1} G_j \]

where \( \Sigma = \sum \sigma_i G_i \). Can be space and time consuming!
- For example, in random regression models with few relatively long time series. In this case simulation is faster.
Simulation study

dub

```r
##     y1  y2 grp subj
## 1  1.70 0  ctrl subj1
## 2  2.01 0  ctrl subj1
## 3  0.65 0  ctrl subj2
## 4  1.39 2  ctrl subj2
## 5  0.31 1  ctrl subj3
## 6  0.94 0  ctrl subj3
## 7  0.55 0  trt1 subj4
## 8  1.20 2  trt1 subj4
## 9  4.49 4  trt1 subj5
##10  4.53 5  trt1 subj5
##11  3.94 2  trt1 subj6
##12  4.02 0  trt1 subj6
```
Task: Test the hypothesis that there is no effect of treatment. How good are the various tests?

Simulate data 1000 times with divine insight: there is no effect of treatment.

Test the hypothesis e.g. at level 5%. If test has correct nominal level we shall reject about 50 times.

If hypothesis is rejected e.g. 100 times then $p$ values are anti-conservative: Effects appear more significant than the really are. That is we draw “too strong” conclusions.

<table>
<thead>
<tr>
<th></th>
<th>0.010</th>
<th>0.050</th>
<th>0.100</th>
</tr>
</thead>
<tbody>
<tr>
<td>lm+X2</td>
<td>0.178</td>
<td>0.282</td>
<td>0.342</td>
</tr>
<tr>
<td>lm+F</td>
<td>0.110</td>
<td>0.240</td>
<td>0.322</td>
</tr>
<tr>
<td>mixed+X2</td>
<td>0.044</td>
<td>0.152</td>
<td>0.240</td>
</tr>
<tr>
<td>mixed+F-KR</td>
<td>0.012</td>
<td>0.044</td>
<td>0.114</td>
</tr>
<tr>
<td>mixed+PB</td>
<td>0.008</td>
<td>0.052</td>
<td>0.108</td>
</tr>
</tbody>
</table>
Motivation: Sugar beets - A split–plot experiment

- Model how sugar percentage in sugar beets depends on harvest time and sowing time.
- Five sowing times ($s$) and two harvesting times ($h$).
- Experiment was laid out in three blocks ($b$).

Experimental plan for sugar beets experiment

```
# Plot allocation:
# Plot          | Block 1         | Block 2         | Block 3         |
# 1-15          | s3 s4 s5 s2 s1  | s3 s2 s4 s5 s1  | s5 s2 s3 s4 s1  |
# 16-30         | s2 s1 s5 s4 s3  | s1 s1 s3 s2 s5  | s1 s4 s3 s2 s5  |
```
beets data

data(beets, package='pbkrtest')
head(beets)

## harvest block sow yield sugpct
## 1 harv1 block1 sow3 128.0 17.1
## 2 harv1 block1 sow4 118.0 16.9
## 3 harv1 block1 sow5  95.0 16.6
## 4 harv1 block1 sow2 131.0 17.0
## 5 harv1 block1 sow1 136.5 17.0
## 6 harv2 block2 sow3 136.5 17.0

par(mfrow=c(1,2))
with(beets, interaction.plot(sow, harvest, sugpct))
with(beets, interaction.plot(sow, harvest, yield))
For simplicity assume no interaction between sowing and harvesting times.

A typical model for such an experiment would be:

\[ y_{hbs} = \mu + \alpha_h + \beta_b + \gamma_s + U_{hb} + \epsilon_{hbs}, \]  \hspace{1cm} (1)

where \( U_{hb} \sim N(0, \omega^2) \) and \( \epsilon_{hbs} \sim N(0, \sigma^2) \).

Notice that \( U_{hb} \) describes the random variation between whole-plots (within blocks).
Using `lmer()` from `lme4` we can test for no effect of sowing and harvest time as:

```r
beet.lg <- lmer(sugpct ~ block + sow + harvest +
                (1 | block:harvest), data=beets, REML=FALSE)
beet.noh <- update(beet.lg, .~. - harvest)
beet.nos <- update(beet.lg, .~. - sow)
```

Both factors appear highly significant

```r
anova(beet.lg, beet.noh) %>% as.data.frame
```

```
## Df  AIC  BIC logLik deviance  Chisq Chi Df Pr(>Chisq)
## beet.noh  9 -69.08 -56.47  43.54 -87.08  NA  NA  NA
## beet.lg  10 -80.00 -65.99  50.00 -100.00 12.91 1  0.0003261
```

```r
anova(beet.lg, beet.nos) %>% as.data.frame
```

```
## Df  AIC  BIC logLik deviance  Chisq Chi Df Pr(>Chisq)
## beet.nos  6 -2.795  5.612  7.398 -14.8 NA  NA  NA
## beet.lg  10 -79.998 -65.986 49.999 -100.0 85.2 4  1.374e-17
```

However, the LRT based $p$–values are anti–conservative: the effect of harvest appears stronger than it is.
As the design is balanced we may make F–tests for each of the effects as:

```r
beets$bh <- with(beets, interaction(block, harvest))
summary(aov(sugpct ~ block + sow + harvest + Error(bh), data=beets))
```

```yaml
## Error: bh
## Df  Sum Sq Mean Sq F value Pr(>F)
## block  2 0.0327 0.0163 2.58   0.28
## harvest 1 0.0963 0.0963 15.21  0.06
## Residuals 2 0.0127 0.0063
##
## Error: Within
## Df Sum Sq Mean Sq   F value Pr(>F)
## sow    4 1.01 0.2525 101 5.7e-13
## Residuals 20 0.05 0.0025
```

Notice: the F–statistics are $F_{1,2}$ for harvest time and $F_{4,20}$ for sowing time.
```r
set.seed("260618")
KRmodcomp(beet.lg, beet.noh)

## F-test with Kenward-Roger approximation; time: 0.13 sec
## large : sugpct ~ block + sow + harvest + (1 | block:harvest)
## small : sugpct ~ block + sow + (1 | block:harvest)
## stat ndf ddf F.scaling p.value
## Ftest 15.2 1.0 2.0 1 0.06

PBmodcomp(beet.lg, beet.noh)

## Bootstrap test; time: 7.94 sec; samples: 1000; extremes: 38;
## large : sugpct ~ block + sow + harvest + (1 | block:harvest)
## small : sugpct ~ block + sow + (1 | block:harvest)
## stat df p.value
## LRT 12.9 1 0.00033
## PBtest 12.9 0.03896

seqPBmodcomp(beet.lg, beet.noh)

## Bootstrap test; time: 8.30 sec; samples: 1000; extremes: 25;
## large : sugpct ~ block + sow + harvest + (1 | block:harvest)
## small : sugpct ~ block + sow + (1 | block:harvest)
## stat df p.value
## LRT 12.9 1 0.00033
## PBtest 12.9 0.02597
```
Final remarks

- Satterthwaite approximation of degrees of freedom on its way in pbkrtest. Faster to compute than Kenward-Roger scales to larger problems.
- pbkrtest available on CRAN
  https://cran.r-project.org/package=pbkrtest
- devel version on github:
  devtools::install_github(hojsgaard/pbkrtest)

Thanks for your attention!