library(Stat5303)
counts <- read.table("kuehl3.dat.txt",header=TRUE)

We have triplicate particle counts on each of three filters from each of two manufacturers for a total of 18 counts.

counts
   manu filter partcount
  1    1    1       1.12
  2    1    1       1.10
  3    1    1       1.12
  4    1    2       0.16
  5    1    2       0.11
  6    1    2       0.26
  7    1    3       0.15
  8    1    3       0.12
  9    1    3       0.12
 10   2    1       0.91
 11   2    1       0.83
 12   2    1       0.95
 13   2    2       0.66
 14   2    2       0.83
 15   2    2       0.61
 16   2    3       2.17
 17   2    3       1.52
 18   2    3       1.58

counts <- within(counts, {manu <- as.factor(manu); filter<-as.factor(filter)})

partcount.lmer <- lmer(partcount ~ 1 + (1|manu/filter),data=counts)

There are a couple of ways to set up the nested model (filter nested in manufacturer). The first is with slash notation: (1|manu/filter) means do the random effects (here just the intercept or constant) for manufacturer and for filter nested in manufacturer. In effect, the (1|manu/filter) expands to (1|manu) + (1|manu:filter).

summary(partcount.lmer)

We get separate estimates for each of the three random effects as usual.

Linear mixed model fit by REML
Formula: partcount ~ 1 + (1 | manu/filter)
   AIC BIC logLik deviance REMLdev
15.41 18.97 -3.705 6.955 7.411
Random effects:
   Groups   Name       Variance Std.Dev.
filter:manu (Intercept) 0.303311  0.55074
   manu (Intercept) 0.103730  0.32207
   Residual         0.025389  0.15934
Number of obs: 18, groups: filter:manu, 6; manu, 2

Fixed effects:
   Estimate Std. Error t value
(Intercept)  0.7956    0.3222   2.469
\texttt{lmer(partcount} \sim 1 + (1|\text{manu}) + (1|\text{manu:filter}), \text{data} = \text{counts})

The other way to do it is to specifically set up the filter (nested in manufacturer) to have a separate effect for each manufacturer by filter combination. This fits the same model and gives the same output.

\texttt{par(mfrow} = c(2, 2))

The lmer.plot command is going to make four plots, one for each explicit random effect and two for residuals. I’m just saying to arrange them 2 by 2.

\texttt{lmer.plot(partcount.lmer)}

And now we see a problem. There is increasing variability in the residuals. I know that the variance of counts can often be stabilized by square roots, so I’ll try that. An alternative is to fit the random model as if it were a fixed model and do Box Cox. That is not “right,” but it should help.
> tmp <- lm(partcount ~ manu/filter, data=counts)

Looking at the residuals from the fixed effects model is not the same thing as looking at the residuals from the random effects model, but it should usually be close enough to be helpful.

> par(mfrow=c(1,1))

Put the graphics back to 1 by 1.

> boxCox(tmp)

Best looks like power .25 or .3, but .5 is well within the interval and is suggested by theory, so we’ll use it.

> rpc.lmer <- lmer(sqrt(partcount) ~ 1 + (1|manu/filter), data=counts)

Let’s try the square root. Note, it is possible to do the transformation within the lmer() statement.
OK, the residuals look better, so we’ll work with the square root counts.

Filter variability is the largest (at about 20 times residual), and manufacturer variability is in between (at about 10 times residual). You would think those would both be very significant, but we only have two manufacturers (one df between), so we have very little information about manufacturer. We will see that manu is not significant.

Linear mixed model fit by REML
Formula: rootpartcount ~ 1 + (1 | manu/filter)
AIC BIC logLik deviance REMLdev
-8.484 -4.922 8.242 -17.78 -16.48
Random effects:
  Groups     Name   Variance Std.Dev.
  filter:manu (Intercept) 0.1054262 0.324694
  manu       (Intercept) 0.0543449 0.233120
  Residual   0.0053046 0.072833
Number of obs: 18, groups: filter:manu, 6; manu, 2

Fixed effects:
  Estimate Std. Error  t value
  (Intercept)  0.8219     0.2122    3.873
To test terms using exactRLRT, we need the full model, the model with only the term of interest, and the full model less the term of interest. Since there are only two terms other than error in the model, the “only the term of interest” models are also the “without the term of interest” models for the other term.

```r
> rpc.filteronly.lmer <- lmer(sqrt(partcount) ~ 1 + (1|manu:filter),data=counts)
> exactRLRT(rpc.filteronly.lmer, rpc.lmer, rpc.manuonly.lmer)
```

Here we test filter, and it is highly significant.

simulated finite sample distribution of RLRT. (p-value based on 10000 simulated values)

data:
RLRT = 27.8483, p-value < 2.2e-16

```r
> exactRLRT(rpc.manuonly.lmer, rpc.lmer, rpc.filteronly.lmer)
```

Here we test manufacturer, and it is not significant. As a point of comparison, these results agree very will with the old school way of doing things.

simulated finite sample distribution of RLRT. (p-value based on 10000 simulated values)

data:
RLRT = 0.4033, p-value = 0.1535

```r
> rpc.mcmc5 <- lmer.mcmc(rpc.lmer, 50000)
```

Let’s get some MCMC samples to look at parameters.
\texttt{par(mfrow=c(4,2))}
\texttt{lmer.mcmc.plots(rpc.mcmc5)}
> lmer.mcmc.plots(rpc.mcmc5, log=TRUE)

The plots look better yet if you plot the log of the ratio of a variance component to the error variance. Filter is fairly stable, and the manufacturer component is just going to zero, with gaps where the value is exactly zero (log of 0 not being defined). Witness the warning (since it only plots every tenth value, almost half of the values are zero).

Warning message:
In xy.coords(x, y, xlabel, ylabel, log) :
  2309 y values <= 0 omitted from logarithmic plot

> lmer.mcmc.intervals(rpc.mcmc5)

Here are the intervals. Zero is in the interval for manufacturer, but it could be big.

<table>
<thead>
<tr>
<th></th>
<th>lower</th>
<th>median</th>
<th>upper</th>
<th>SE</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>0.373615419</td>
<td>0.8177015e-01</td>
<td>1.21274913</td>
<td>0.210405186</td>
</tr>
<tr>
<td>filter:manu</td>
<td>0.043123936</td>
<td>1.437820e-01</td>
<td>0.62863391</td>
<td>0.152003545</td>
</tr>
<tr>
<td>manu</td>
<td>0.000000000</td>
<td>2.393180e-07</td>
<td>0.12465288</td>
<td>0.044364959</td>
</tr>
<tr>
<td>sigma2</td>
<td>0.002691776</td>
<td>5.779265e-03</td>
<td>0.01475100</td>
<td>0.003142901</td>
</tr>
</tbody>
</table>
We can also get intervals for functional combinations of these variables. For example, suppose that we want the correlation between two observations on the same filter. These observations would share the same random manufacturer and filter random effects ($\alpha_i$ and $\beta_{j(i)}$, but they would have their own error $\epsilon_{ijk}$. Thus the correlation would be $(\sigma^2_\alpha + \sigma^2_\beta)/(\sigma^2_\alpha + \sigma^2_\beta + \sigma^2)$.

What we do is just produce the same functional combination of the variables from the MCMC output, and then find the middle 95 (or whatever) percent.

```r
> m <- rpc.mcmc5$mcmcout
```

There is a component in the output of `lmer.mcmc()` called `mcmcout`. The component is a matrix with one row for every tenth MCMC trial and one column for every parameter.

```r
> dim(m)
```

We had 50,000 MCMC reps, so 5,000 rows in `m`. There are four parameters (the intercept and three variance components `filter:manu`, `manu`, and `sigma2`), so `m` has four columns.

```r
> colnames(rpc.mcmc5$mcmcout)
```

If you need a reminder about the order of the columns, look at the column labels.

```r
> ratio <- (m[,2]+m[,3])/(m[,2]+m[,3]+m[,4])
```

The ratio we are looking for is the sum of the second and third columns (`filter:manu` and `manu`) divided by the sum of the last three columns (all the variance components).

```r
> .025*5000
```

For a 95% confidence interval we’re going to want to go in 2.5% on each end, so we’ll want the 125th smallest and the 125th biggest ratio.

```r
> ratio.sorted <- sort(ratio)
```

Sort the ratios.

```r
> ratio.sorted[c(125,5000-125)]
```

Get 125th biggest and smallest for our interval, which runs from .88 to .996.

As it happens, the smallest ratio occurred on simulation 2361, and the largest on simulation 35951, but that is not very useful.

```r
> rpcb.lmer <- lmer(sqrt(partcount) ~ manu + (1|manu:filter), data=counts)
```

What if there really are only two manufacturers? In that case we are hardly taking a random sample, we are looking at all of them. We should now fit manufacturer as a fixed effect, but filters would still be random nested in manufacturer.
> summary(rpcb.lmer)

Note that we have a fixed coefficient for the first manufacturer, and only random effects for filter and residuals. In this nicely balanced case the error and filter estimated effects are the same as in the fully random model.

Linear mixed model fit by REML
Formula: rootpartcount ~ manu + (1 | manu:filter)
   AIC  BIC logLik deviance REMLdev
-8.221 -4.66  8.11  -20.72  -16.22
Random effects:
  Groups   Name     Variance     Std.Dev.
  manu:filter (Intercept)  0.1054262  0.324694
  Residual             0.0053046  0.072833
Number of obs: 18, groups: manu:filter, 6

Fixed effects:
   Estimate Std. Error t value
  (Intercept)  0.8219     0.1337   6.149
  manu1      -0.2122     0.1337  -1.588

Correlation of Fixed Effects:
    (Intr)
manu1 0.000

> anova(rpcb.lmer)

anova() for an lmer model with fixed effects gives an F test statistic for each fixed term. Unfortunately, it doesn’t give a denominator df and thus gives no p-value. We would get a p-value if we had used lme() instead of lmer(). The issue is that it is easier to get an appropriate denominator degrees of freedom for these tests when there is only nesting of random effects. lme() only does nesting, so it gives a denominator df. lmer() does crossing as well as nesting, and lmer() does not try to get a denominator df, even in cases where there is only nesting.

Analysis of Variance Table
   Df Sum Sq Mean Sq F value
  manu  1 0.01337 0.013373  2.5209

> rpcb.mcmc5 <- lmer.mcmc(rpcb.lmer,50000)

Do the MCMC.

> lmer.mcmc.anova(rpcb.mcmc5)

I have written a function that will take the MCMC output and do an “anova” on the fixed effect. In the “old school” analysis, the t-value for manu1 we saw above (-1.588) would have 4 degrees of freedom. Computing a two-sided p-value with that df gives us 0.19, which is just about what we got here.

chisq Df MC p-value
  (Intercept)  18.818904 1  0.0034
  manu         1.315096 1  0.2018

> lmer.KR.anova(rpcb.lmer)

I also have another testing method for fixed effects that uses some approximations due to Kenward and Rogers. KR was designed to give approximate results that are familiar and comfortable and will usually agree with “old school” approaches in simple cases.

    F   df1  df2   p-value
manu  2.520925 1  4  0.1875383
Let’s look at what we can do using lme instead of lmer. We’ll first get rid of the lme4 package, then we’ll load in the nlme package. (I think lme4 and nlme sometimes step on each others toes.)

We will for the moment treat manufacturer as fixed. lme() is happy to work with nested random factors, but it won’t cross them. So in order to get something like manu:filter we need to combine it directly. The join function makes a new factor combining all the levels of two factors.

When using lme you first describe the fixed parts like in a fixed only model, and then you put the random parts in a separate argument. Note that in the random= argument there is no “response” on the left hand side of the tilde. Don’t ask me why.

The parameter estimates and other outputs are the same here as they were for the lmer model, but lme is willing to give a p-value for fixed effects.

```
Linear mixed-effects model fit by REML
Data: NULL
AIC BIC logLik
-8.221371 -5.131016 8.110685

Random effects:
Formula: ~1 | filterinmanu
 (Intercept) Residual
StdDev: 0.324694 0.07283274

Fixed effects: rootpartcount ~ manu
 (Intercept) manu
Value Std.Error DF t-value p-value
(Intercept) 0.821899 0.1336628 12 6.149050 0.0000
manu1 -0.212222 0.1336628 4 -1.587742 0.1875

Correlation:
 (Intr)
manu1 0

Standardized Within-Group Residuals:
 Min Q1 Med Q3 Max
-1.1728581 -0.4687896 -0.1102852 0.2743776 2.1534105

Number of Observations: 18
Number of Groups: 6
```

lme() is willing to have a go at p-values for testing fixed terms (but recall that it only handles nested chains of random terms).

```
numDF denDF F-value p-value
(Intercept) 1 12 37.81081 <.0001
manu1 1 4 2.52093 0.1875
```
> detach("package:nlme"); attach(lme4)
  Let's get back to lme4.
> soils <- read.table("kuehl5.dat.txt", header=TRUE)
  Data from problem 5-7 of Kuehl (1994 Duxbury). Fifteen fields are chosen at random. Two
  subsections are chosen at random from each field. Soil porosity is measured at a random
  location of each subsection; some subsections are measured at two locations.
  Field and subsection are random, with subsection nested in field. The data set is not bal-
  anced.
> soils <- within(soils, {field <- as.factor(field);
section <- as.factor(section); sect <- as.factor(sect)})
> soils
  Note that the variable section enumerates all the sections, but the variable sect is 1 or 2
  within each field.
<table>
<thead>
<tr>
<th>field</th>
<th>section</th>
<th>sect</th>
<th>porosity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3.846</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>3.712</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>2</td>
<td>5.629</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>5.629</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>1</td>
<td>5.087</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>2</td>
<td>4.621</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>5</td>
<td>4.411</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>6</td>
<td>3.357</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>7</td>
<td>3.991</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>8</td>
<td>5.766</td>
</tr>
<tr>
<td>11</td>
<td>5</td>
<td>9</td>
<td>5.677</td>
</tr>
<tr>
<td>12</td>
<td>5</td>
<td>10</td>
<td>3.333</td>
</tr>
<tr>
<td>13</td>
<td>6</td>
<td>11</td>
<td>4.355</td>
</tr>
<tr>
<td>14</td>
<td>6</td>
<td>11</td>
<td>6.292</td>
</tr>
<tr>
<td>15</td>
<td>6</td>
<td>12</td>
<td>4.940</td>
</tr>
<tr>
<td>16</td>
<td>6</td>
<td>12</td>
<td>4.810</td>
</tr>
<tr>
<td>17</td>
<td>7</td>
<td>13</td>
<td>2.983</td>
</tr>
<tr>
<td>18</td>
<td>7</td>
<td>14</td>
<td>4.396</td>
</tr>
<tr>
<td>19</td>
<td>8</td>
<td>15</td>
<td>5.603</td>
</tr>
<tr>
<td>20</td>
<td>8</td>
<td>16</td>
<td>3.683</td>
</tr>
<tr>
<td>21</td>
<td>9</td>
<td>17</td>
<td>5.942</td>
</tr>
<tr>
<td>22</td>
<td>9</td>
<td>18</td>
<td>5.014</td>
</tr>
<tr>
<td>23</td>
<td>10</td>
<td>19</td>
<td>5.143</td>
</tr>
<tr>
<td>24</td>
<td>10</td>
<td>20</td>
<td>4.061</td>
</tr>
<tr>
<td>25</td>
<td>11</td>
<td>21</td>
<td>3.835</td>
</tr>
<tr>
<td>26</td>
<td>11</td>
<td>21</td>
<td>2.964</td>
</tr>
<tr>
<td>27</td>
<td>11</td>
<td>22</td>
<td>4.584</td>
</tr>
<tr>
<td>28</td>
<td>11</td>
<td>22</td>
<td>4.398</td>
</tr>
<tr>
<td>29</td>
<td>12</td>
<td>23</td>
<td>4.193</td>
</tr>
<tr>
<td>30</td>
<td>12</td>
<td>24</td>
<td>4.125</td>
</tr>
<tr>
<td>31</td>
<td>13</td>
<td>25</td>
<td>3.074</td>
</tr>
<tr>
<td>32</td>
<td>13</td>
<td>26</td>
<td>3.483</td>
</tr>
<tr>
<td>33</td>
<td>14</td>
<td>27</td>
<td>3.867</td>
</tr>
<tr>
<td>34</td>
<td>14</td>
<td>28</td>
<td>4.212</td>
</tr>
<tr>
<td>35</td>
<td>15</td>
<td>29</td>
<td>6.247</td>
</tr>
<tr>
<td>36</td>
<td>15</td>
<td>30</td>
<td>4.730</td>
</tr>
</tbody>
</table>
```r
> fit1 <- lmer(porosity ~ 1 + (1|field/sect), data=soils)
  Fit the model with sect nested within field.
> fit2 <- lmer(porosity ~ 1 + (1|field) + (1|section), data=soils)
  Fit the model with field and section. Since section enumerates all the sections individually,
  we don’t need to do “nesting” in the model. We’ll get the same results.
> fit1
There is some evidence of field to field variability, but the section within field is estimated
at zero.
Linear mixed model fit by REML
Formula: porosity ~ 1 + (1 | field/sect)
  AIC  BIC   logLik deviance REMLdev
110.6 116.9 -51.28 100.9   102.6
Random effects:
  Groups   Name    Variance Std.Dev.
  sect:field (Intercept) 0.000000 0.00000
  field     (Intercept) 0.059483 0.24389
  Residual              0.935989 0.96747
Number of obs: 36, groups: sect:field, 30; field, 15
Fixed effects:
  Estimate Std. Error  t value
  (Intercept) 4.4037    0.1741   25.29
> fit2
As hoped, same results this way.
Linear mixed model fit by REML
Formula: porosity ~ 1 + (1 | field) + (1 | section)
  AIC  BIC   logLik deviance REMLdev
110.6 116.9 -51.28 100.9   102.6
Random effects:
  Groups   Name    Variance Std.Dev.
  section  (Intercept) 0.000000 0.00000
  field     (Intercept) 0.059483 0.24389
  Residual              0.935989 0.96747
Number of obs: 36, groups: section, 30; field, 15
Fixed effects:
  Estimate Std. Error  t value
  (Intercept) 4.4037    0.1741   25.29
> fit.nofield <- lmer(porosity ~ 1 + (1|field:sect), data=soils)
> fit.onlyfield <- lmer(porosity ~ 1 + (1|field), data=soils)
> exactRLRT(fit.onlyfield, fit1, fit.nofield)
Test of field effect is not significant.
  simulated finite sample distribution of RLRT. (p-value based
  on 10000 simulated values)
  data:
  RLRT = 0.1197, p-value = 0.3388
```
> fit1.mcmc <- lmer.mcmc(fit1,50000)
> lmer.mcmc.plots(fit1.mcmc,log=TRUE)

Even 50,000 is not enough for this to have settled down entirely. Lot’s of zeros showing up.

Warning messages:
1: In xy.coords(x, y, xlabel, ylabel, log) :
   2524 y values <= 0 omitted from logarithmic plot
2: In xy.coords(x, y, xlabel, ylabel, log) :
   2348 y values <= 0 omitted from logarithmic plot
\texttt{lmer.mcmc.intervals(fit1.mcmc)}

When we look at the intervals, they show the section and field variances not significantly different from zero.

<table>
<thead>
<tr>
<th>lower</th>
<th>median</th>
<th>upper</th>
<th>SE</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept) 4.0429291</td>
<td>4.397537e+00</td>
<td>4.75294137</td>
<td>0.17593126</td>
</tr>
<tr>
<td>sect:field 0.0000000</td>
<td>0.000000e+00</td>
<td>0.08718555</td>
<td>0.03057076</td>
</tr>
<tr>
<td>field      0.0000000</td>
<td>1.497799e-10</td>
<td>0.03423929</td>
<td>0.01448301</td>
</tr>
<tr>
<td>sigma2     0.6527158</td>
<td>1.014978e+00</td>
<td>1.69588286</td>
<td>0.26718021</td>
</tr>
</tbody>
</table>

\texttt{par(mfrow=c(2,2))}
\texttt{lmer.plot(fit1)}

What the heck?!! Look at that last plot. It looks all the world like the field effects didn’t really pick up all of the field to field variation, because there is still some trend in the residuals.

Well, guess what? In some cases, such as this one, random effects can be “shrunk” back towards zero. This happens more when the variance of the random effect is small compared to the error variance. In this case, residual variance is much larger than field variance, and field variance and and its random effects are being shrunk towards zero. This shows up in the residual plot as trend: negative random effects aren’t negative enough (at least by what we’re used to) and positive random effects aren’t positive enough.
> fit3 <- lmer(porosity ~ field + (1|section), data=soils)

OK, now suppose that we are only worried about these 15 fields, and we’re not thinking of
them as sampled from some larger population. In this case, we would treat field as fixed.
Note, we could also use (1|field:sect) for the random term.

> fit3

Section variance is still estimated at zero, but we now have individual estimates for the
fields.

Linear mixed model fit by REML
Formula: porosity ~ field + (1 | section)
   AIC  BIC logLik deviance REMLdev
110.8 137.7  -38.41  81.6  76.81
Random effects:
  Groups   Name   Variance Std.Dev.
   section (Intercept) 0.00000  0.000
       Residual     0.96827  0.984
Number of obs: 36, groups: section, 30

Fixed effects:
                         Estimate Std. Error t value
(Intercept)                4.42307   0.17043  25.952
field1                    -0.62107   0.48871  -1.271
field2                    -0.43093   0.66980  -0.643
field3                    -0.53907   0.66980  -0.805
field4                     0.45543   0.66980   0.680
field5                     0.08193   0.66980   0.122
field6                    -0.67618   0.48871  -1.384
field7                    -0.73357   0.66980  -1.095
field8                     0.21993   0.66980   0.328
field9                    -1.05493   0.66980  -1.575
field10                   -0.17893   0.66980  -0.267
field11                    0.47782   0.48871   0.978
field12                   -0.26407   0.66980  -0.394
field13                   -1.14457   0.66980  -1.709
field14                   -0.38357   0.66980  -0.573
...

> fit2@ranef

Although we don’t usually look at the actual estimated random effects, we can get them.
For fit2, the first 30 (for section) are all zero, then we see the next 15 for field. Note that our
best guess of these random effects is considerably smaller than what we would estimate if
they were fixed effects (which we can see in the output of fit3).

[1]  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000
[6]  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000
[11] 0.00000000  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000
[16] 0.00000000  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000
[21] 0.00000000  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000
[26] 0.00000000  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000
[31] -0.12193305  0.05077171  -0.05859339  0.05353402  0.01142282
[36]  0.14095733  -0.08052279   0.02698198  0.12112617  0.02235933
[41] -0.09290314  -0.02758782  -0.12686202  -0.04106115  0.12231002
> par(mfrow=c(2,2))
> lmer.plot(fit3)

These residuals are more like what we expect, because the fixed effects are not being shrunk back like we were seeing when we treated field as a random effect.

Normal QQ plot of section effects

Normal QQ plot of Residuals

> fit3.mcmc <- lmer.mcmc(fit3,50000)
> lmer.mcmc.plots(fit3.mcmc,log=TRUE)

We’ll not show these here, as there are 34 plots!
> `lmer.mcmc.intervals(fit3.mcmc)`

We’re seeing just a shade more variability here than in the output of fit3, but it’s not much of a difference.

<table>
<thead>
<tr>
<th></th>
<th>lower</th>
<th>median</th>
<th>upper</th>
<th>SE</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>4.0601051</td>
<td>4.4337112</td>
<td>4.7980495</td>
<td>0.1861582</td>
</tr>
<tr>
<td>field1</td>
<td>-1.6911292</td>
<td>-0.6149087</td>
<td>0.3774820</td>
<td>0.5221897</td>
</tr>
<tr>
<td>field2</td>
<td>-0.9437626</td>
<td>0.3796817</td>
<td>1.7097702</td>
<td>0.6722641</td>
</tr>
<tr>
<td>field3</td>
<td>-1.8665268</td>
<td>-0.4635144</td>
<td>0.9300701</td>
<td>0.7197178</td>
</tr>
<tr>
<td>field4</td>
<td>-1.1352510</td>
<td>0.4423249</td>
<td>1.8682280</td>
<td>0.7389921</td>
</tr>
<tr>
<td>field5</td>
<td>-1.2204997</td>
<td>0.1267372</td>
<td>1.5236367</td>
<td>0.6827920</td>
</tr>
<tr>
<td>field6</td>
<td>-0.3341452</td>
<td>0.6711747</td>
<td>1.6577772</td>
<td>0.5098678</td>
</tr>
<tr>
<td>field7</td>
<td>-2.0576945</td>
<td>-0.6869103</td>
<td>0.6575987</td>
<td>0.6880298</td>
</tr>
<tr>
<td>field8</td>
<td>-1.2219240</td>
<td>0.2083560</td>
<td>1.6960486</td>
<td>0.7245083</td>
</tr>
<tr>
<td>field9</td>
<td>-0.4384252</td>
<td>1.0215854</td>
<td>2.5466026</td>
<td>0.7242237</td>
</tr>
<tr>
<td>field10</td>
<td>-1.3230680</td>
<td>0.1579496</td>
<td>1.4900398</td>
<td>0.7114554</td>
</tr>
<tr>
<td>field11</td>
<td>-1.4628051</td>
<td>-0.4862379</td>
<td>0.5525603</td>
<td>0.5087946</td>
</tr>
<tr>
<td>field12</td>
<td>-1.6631675</td>
<td>-0.2712383</td>
<td>1.1628171</td>
<td>0.7115539</td>
</tr>
<tr>
<td>field13</td>
<td>-2.6236514</td>
<td>-1.1488082</td>
<td>0.2532431</td>
<td>0.7261034</td>
</tr>
<tr>
<td>field14</td>
<td>-1.8467741</td>
<td>-0.3947314</td>
<td>0.9866249</td>
<td>0.7059231</td>
</tr>
<tr>
<td>section</td>
<td>0.0000000</td>
<td>0.0000000</td>
<td>0.2154743</td>
<td>0.1101968</td>
</tr>
<tr>
<td>sigma2</td>
<td>0.5640519</td>
<td>0.9856555</td>
<td>2.0049233</td>
<td>0.3773852</td>
</tr>
</tbody>
</table>

> `lmer.mcmc.anova(fit3.mcmc)`

Here’s how we test fixed effects. Field is not significant.

<table>
<thead>
<tr>
<th></th>
<th>chisq</th>
<th>Df</th>
<th>MC</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>564.52447</td>
<td>1</td>
<td>0.0000</td>
<td></td>
</tr>
<tr>
<td>field</td>
<td>13.33335</td>
<td>14</td>
<td>0.4466</td>
<td></td>
</tr>
</tbody>
</table>

> `lmer.KR.anova(fit3)`

Alternative test for fixed effects, again not significant.

<table>
<thead>
<tr>
<th></th>
<th>F</th>
<th>df1</th>
<th>df2</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>1.186108</td>
<td>14</td>
<td>4.334391</td>
<td>0.4717713</td>
</tr>
</tbody>
</table>
These are the data from Table 7.10 of Kuehl. We are investigating how well an instrument measures serum glucose. We measure at three fixed levels of glucose. The machine may work differently on different days, so we choose three days at random. Also, we will do two runs or batches on the machine each day. That is, do everything once, and then do it all over again. There may be a run effect nested in day. Finally, each concentration is measured twice during each run.

```r
> glucose <- read.table("kuehl4.dat.txt", header=TRUE)
```

Concentration is fixed, and day and run are random. Run is nested in day, and everything else crosses.

```r
> glucose <- within(glucose,
+ {
+ conc <- factor(conc);
+ day <- factor(day); run <- factor(rn)
+ })
```

```r
> glu.lmer <- lmer(y ~ conc + (1|day/run) + (1|conc:day) + (1|conc:day:run))
```
Hmmm, perhaps we should take the square root of these data.

So let’s redo with square root concentration.

This looks a bit better.
We have little evidence that day has any effect, or that there is a difference in days by concentration. Concentration is significant (well, we haven’t done the “right” test, but it’s going to be significant). Run seems to have an effect, as does concentration by run. So every time we set up the apparatus we get a different answer, and the difference varies a lot by the concentration we happen to be measuring. This is not a reassuring finding.

```r
> rglu.lmer
```

Linear mixed model fit by REML
Formula: sqrt(y) ~ conc + (1 | day/run) + (1 | conc:day) + (1 | conc:day:run)
   AIC BIC logLik deviance REMLdev
-23.45 -10.78 19.72 -51.07 -39.45
Random effects:
Groups Name Variance Std.Dev.
conc:day:run (Intercept) 0.0238342 0.154383
cconc:day (Intercept) 0.0050494 0.071059
crun:day (Intercept) 0.0092985 0.096429
day (Intercept) 0.0000000 0.000000
Residual 0.0031903 0.056483
Number of obs: 36, groups: conc:day:run, 18; conc:day, 9; run:day, 6; day, 3

Fixed effects:
   Estimate Std. Error t value
(Intercept) 10.43086  0.05936  175.72
conc1  -3.93972  0.06283  -62.70
conc2   1.25351  0.06283   19.95
Correlation of Fixed Effects:

<table>
<thead>
<tr>
<th>Effect</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intr)</td>
<td></td>
</tr>
<tr>
<td>conc1</td>
<td>0.000</td>
</tr>
<tr>
<td>conc2</td>
<td>-0.500</td>
</tr>
</tbody>
</table>

```r
gerlu.norun <- lmer(sqrt(y) ~ conc + (1|day) + (1|conc:day) + (1|conc:day:run),
data=glucose)
gerlu.runonly <- lmer(sqrt(y)~conc+(1|day:run),data=glucose)
> exactRLRT(rglu.runonly,rglu.lmer,rglu.norun)

Run is not significant.
```

Simulated finite sample distribution of RLRT. (p-value based on 10000 simulated values)

```r
data:
RLRT = 0.6062, p-value = 0.1763
```

```r
gerlu.noconcrun <- lmer(sqrt(y) ~ conc + (1|day/run) + (1|conc:day),data=glucose)
gerlu.concrunonly <- lmer(sqrt(y) ~ conc + (1|conc:day:run),data=glucose)
> exactRLRT(glu.concrunonly, glu.lmer,glu.noconcrun)

Concentration by run is very significant.
```

Simulated finite sample distribution of RLRT. (p-value based on 10000 simulated values)

```r
data:
RLRT = 25.1997, p-value < 2.2e-16
```

```r
> rglu.mcmc <- lmer.mcmc(rglu.lmer,20000)
20000 may be a bit few.
> lmer.mcmc.intervals(rglu.mcmc)
Run by concentration is the only one that doesn’t seem small compared to error, which fits
with our tests.

<table>
<thead>
<tr>
<th>Effect</th>
<th>lower</th>
<th>median</th>
<th>upper</th>
<th>SE</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>10.331</td>
<td>1.0433</td>
<td>1.0539</td>
<td>0.0529</td>
</tr>
<tr>
<td>conc1</td>
<td>-4.078</td>
<td>-3.933</td>
<td>-3.804</td>
<td>0.069</td>
</tr>
<tr>
<td>conc2</td>
<td>1.118</td>
<td>1.2489</td>
<td>1.3801</td>
<td>0.066</td>
</tr>
<tr>
<td>conc:day:run</td>
<td>0.0151</td>
<td>3.5488</td>
<td>9.3338</td>
<td>5.9444</td>
</tr>
<tr>
<td>conc:day</td>
<td>0.0000</td>
<td>0.0000</td>
<td>1.1709</td>
<td>0.4783</td>
</tr>
<tr>
<td>run:day</td>
<td>0.0000</td>
<td>3.5591</td>
<td>1.0756</td>
<td>2.3323</td>
</tr>
<tr>
<td>day</td>
<td>0.0000</td>
<td>8.2532</td>
<td>2.3323</td>
<td>1.6293</td>
</tr>
<tr>
<td>sigma2</td>
<td>0.0018</td>
<td>3.3099</td>
<td>8.4195</td>
<td>1.6293</td>
</tr>
</tbody>
</table>

> lmer.mcmc.anova(rglu.mcmc)
As anticipated, the concentrations differ.

<table>
<thead>
<tr>
<th>chisq</th>
<th>Df</th>
<th>MC</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>38934.758</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>conc</td>
<td>3456.903</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

> lmer.KR.anova(rglu.lmer)
Kenward and Rogers approximation agrees.

<table>
<thead>
<tr>
<th>F</th>
<th>df1</th>
<th>df2</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>conc</td>
<td>1720.584</td>
<td>5</td>
<td>6.04631</td>
</tr>
</tbody>
</table>