

# **lmer for SAS PROC MIXED Users**

Douglas Bates  
Department of Statistics  
University of Wisconsin – Madison  
Bates@wisc.edu

## **1 Introduction**

The `lmer` function from the `lme4` package for R is used to fit linear mixed-effects models. It is similar in scope to the SAS procedure PROC MIXED described in Littell et al. (1996).

A file on the SAS Institute web site (<http://www.sas.com>) contains all the data sets in the book and all the SAS programs used in Littell et al. (1996). We have converted the data sets from the tabular representation used for SAS to the `data.frame` objects used by `lmer`. To help users familiar with SAS PROC MIXED get up to speed with `lmer` more quickly, we provide transcripts of some `lmer` analyses paralleling the SAS PROC MIXED analyses in Littell et al. (1996).

In this paper we highlight some of the similarities and differences of `lmer` analysis and SAS PROC MIXED analysis.

## **2 Similarities between lmer and SAS PROC MIXED**

Both SAS PROC MIXED and `lmer` can fit linear mixed-effects models expressed in the Laird-Ware formulation. For a single level of grouping Laird and Ware (1982) write the  $n_i$ -dimensional response vector  $\mathbf{y}_i$  for the  $i$ th experimental

unit as

$$\begin{aligned} \mathbf{y}_i &= \mathbf{X}_i\boldsymbol{\beta} + \mathbf{Z}_i\mathbf{b}_i + \boldsymbol{\epsilon}_i, \quad i = 1, \dots, M \\ \mathbf{b}_i &\sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}), \quad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}) \end{aligned} \tag{1}$$

where  $\boldsymbol{\beta}$  is the  $p$ -dimensional vector of *fixed effects*,  $\mathbf{b}_i$  is the  $q$ -dimensional vector of *random effects*,  $\mathbf{X}_i$  (of size  $n_i \times p$ ) and  $\mathbf{Z}_i$  (of size  $n_i \times q$ ) are known fixed-effects and random-effects regressor matrices, and  $\boldsymbol{\epsilon}_i$  is the  $n_i$ -dimensional *within-group error* vector with a spherical Gaussian distribution. The assumption  $\text{Var}(\boldsymbol{\epsilon}_i) = \sigma^2\mathbf{I}$  can be relaxed using additional arguments in the model fitting.

The basic specification of the model requires a linear model expression for the fixed effects and a linear model expression for the random effects. In SAS PROC MIXED the fixed-effects part is specified in the `model` statement and the random-effects part in the `random` statement. In `lmer` the fixed effects and the random effects are both specified as terms in the `formula` argument to `lmer`.

Both SAS PROC MIXED and `lmer` allow a mixed-effects model to be fit by maximum likelihood (`method = ml` in SAS) or by maximum residual likelihood, sometimes also called restricted maximum likelihood or REML. This is the default criterion in SAS PROC MIXED and in `lmer`. To get ML estimates use the optional argument `REML=FALSE` in the call to `lmer`.

### 3 Important differences

The output from PROC MIXED typically includes values of the Akaike Information Criterion (AIC) and Schwartz's Bayesian Criterion (SBC). These are used to compare different models fit to the same data. The output of the `summary` function applied to the object created by `lmer` also produces values of AIC and BIC but the definitions used in older versions of PROC MIXED are different from those used in more recent versions of PROC MIXED and in `lmer`. In `lmer` the definitions are such that "smaller is better". In some older versions of PROC MIXED the definitions are such that "bigger is better".

When models are fit by REML, the values of AIC, SBC (or BIC) and the log-likelihood can only be compared between models with exactly the same fixed-effects structure. When models are fit by maximum likelihood these criteria can be compared between any models fit to the same data. That is,

these quality-of-fit criteria can be used to evaluate different fixed-effects specifications or different random-effects specifications or different specifications of both fixed effects and random effects.

We encourage developing and testing the model using likelihood ratio tests or the AIC and BIC criteria. Once a form for both the random effects and the fixed effects has been determined, the model can be refit with `REML = TRUE` if the restricted estimates of the variance components are desired. Note that the `update` function provides a convenient way of refitting a model with changes to one or more arguments.

## 4 Data manipulation

Both PROC MIXED and `lmer` work with data in a tabular form with one row per observation. There are, however, important differences in the internal representations of variables in the data.

In SAS a qualitative factor can be stored either as numerical values or alphanumeric labels. When a factor stored as numerical values is used in PROC MIXED it is listed in the `class` statement to indicate that it is a factor. In S this information is stored with the data itself by converting the variable to a factor when it is first stored. If the factor represents an ordered set of levels, it should be converted to an `ordered` factor.

For example the SAS code

```
data animal;
  input trait animal y;
  datalines;
1 1 6
1 2 8
1 3 7
2 1 9
2 2 5
2 3 .
;
```

would require that the `trait` and `animal` variables be specified in a `class` statement in any model that is fit.

In R these data could be read from a file, say `animal.dat`, and converted to factors by

```
animal <- within(read.table("animal.dat", header = TRUE),
  {
```

```

        trait <- factor(trait)
        animal <- factor(animal)
    })

```

In general it is a good idea to check the types of variables in a data frame before working with it. One way of doing this is to apply the function `data.class` to each variable in turn using the `sapply` function.

```

> sapply(Animal, data.class)
      Sire      Dam AvgDailyGain
"factor"  "factor"  "numeric"
> str(Animal)
'data.frame':      20 obs. of  3 variables:
 $ Sire      : Factor w/  5 levels "1","2","3","4",...: 1 1 1 1 2 2 2 2 3 3 ...
 $ Dam       : Factor w/  2 levels "1","2": 1 1 2 2 1 1 2 2 1 1 ...
 $ AvgDailyGain: num  2.24 1.85 2.05 2.41 1.99 1.93 2.72 2.32 2.33 2.68 ...
- attr(*, "ginfo")=List of 7
 ..$ formula      :Class 'formula' language AvgDailyGain ~ 1 | Sire/Dam
 .. .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
 ..$ order.groups:List of 2
 .. ..$ Sire: logi TRUE
 .. ..$ Dam : logi TRUE
 ..$ FUN          :function (x)
 ..$ outer        : NULL
 ..$ inner        : NULL
 ..$ labels       :List of 1
 .. ..$ AvgDailyGain: chr "Average Daily Weight Gain"
 ..$ units        : list()

```

## 4.1 Unique levels of factors

Designs with nested grouping factors are indicated differently in the two languages. An example of such an experimental design is the semiconductor experiment described in section 2.2 of Littell et al. (1996) where twelve wafers are assigned to four experimental treatments with three wafers per treatment. The levels for the wafer factor are 1, 2, and 3 but the wafer factor is only meaningful within the same level of the treatment factor, *et*. There is nothing associating wafer 1 of the third treatment group with wafer 1 of the first treatment group.

In SAS this nesting of factors is denoted by `wafer(et)`. In S the nesting is written with `ET/Wafer` and read “wafer within ET”. If both levels of

nested factors are to be associated with random effects then this is all you need to know. You would use an expression with a "/" in the grouping factor part of the formula in the call to `lmer` object. The random effects term would be either

```
(1 | ET/Wafer)
```

or, equivalently

```
(1 | ET:Wafer) + (1 | ET)
```

In this case, however, there would not usually be any random effects associated with the “experimental treatment” or `ET` factor. The only random effects are at the `Wafer` level. It is necessary to create a factor that will have unique levels for each `Wafer` within each level of `ET`. One way to do this is to assign

```
> Semiconductor <- within(Semiconductor, Grp <- factor(ET:Wafer))
```

after which we could specify a random effects term of `(1 | Grp)`. Alternatively, we can use the explicit term

```
(1 | ET:Wafer)
```

## 4.2 General approach

As a general approach to importing data into R for mixed-effects analysis you should:

- Create a `data.frame` with one row per observation and one column per variable.
- Use `factor` or `as.factor` to explicitly convert any ordered factors to class `ordered`.
- Use `ordered` or `as.ordered` to explicitly convert any ordered factors to class `ordered`.
- If necessary, use interaction terms to create a factor with unique levels from inner nested factors.
- Plot the data. Plot it several ways. The use of lattice graphics is closely integrated with the `lme4` library. Lattice plots can provide invaluable insight into the structure of the data. Use them.

## 5 Contrasts

When comparing estimates produced by SAS PROC MIXED and by `lmer` one must be careful to consider the contrasts that are used to define the effects of factors. In SAS a model with an intercept and a qualitative factor is defined in terms of the intercept and the indicator variables for all but the last level of the factor. The default behaviour in S is to use the Helmert contrasts for the factor. On a balanced factor these provide a set of orthogonal contrasts. In R the default is the “treatment” contrasts which are almost the same as the SAS parameterization except that they drop the indicator of the first level, not the last level.

When in doubt, check which contrasts are being used with the `contrasts` function.

To make comparisons easier, you may find it worthwhile to declare  
> `options(contrasts = c(factor = "contr.SAS", ordered = "contr.poly"))`

at the beginning of your session.

## References

Nan M. Laird and James H. Ware. Random-effects models for longitudinal data. *Biometrics*, 38:963–974, 1982.

Ramon C. Littell, George A. Milliken, Walter W. Stroup, and Russell D. Wolfinger. *SAS System for Mixed Models*. SAS Institute, Inc., 1996.

## A AvgDailyGain

```
> print(xyplot(adg ~ Treatment | Block, AvgDailyGain, type = c("g", "p", "r")
+         xlab = "Treatment (amount of feed additive)",
+         ylab = "Average daily weight gain (lb.)", aspect = "xy",
+         index.cond = function(x, y) coef(lm(y ~ x))[1]))

> ## compare with output 5.1, p. 178
> (fmlAdg <- lmer(adg ~ (Treatment - 1)*InitWt + (1 | Block), AvgDailyGain))
Linear mixed model fit by REML ['lmerMod']
Formula: adg ~ (Treatment - 1) * InitWt + (1 | Block)
Data: AvgDailyGain
```

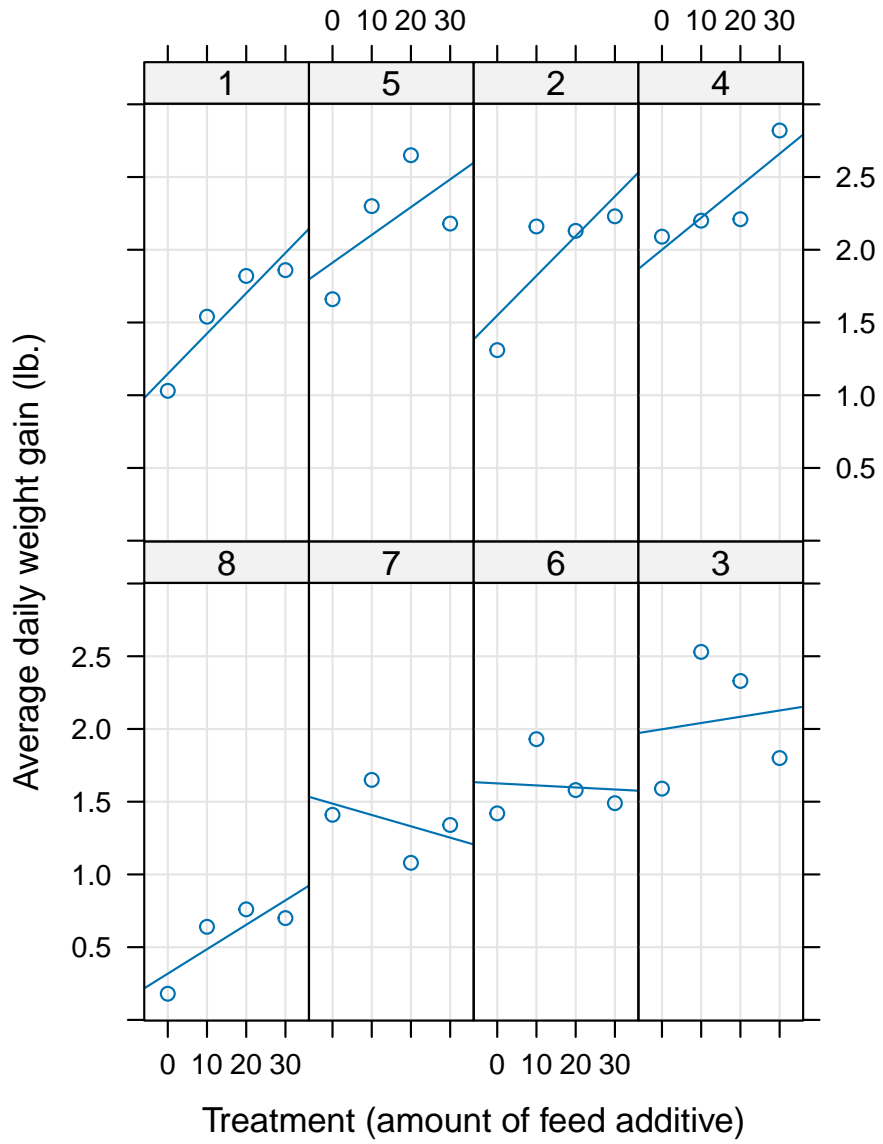


Figure 1: Average daily weight gain

REML criterion at convergence: 65.3268

Random effects:

Groups	Name	Std.Dev.
Block	(Intercept)	0.5092
Residual		0.2223

Number of obs: 32, groups: Block, 8

Fixed Effects:

Treatment0	Treatment10	Treatment20
0.439137	1.426119	0.479629
Treatment30	InitWt	Treatment0:InitWt
0.200107	0.004448	-0.002154
Treatment10:InitWt	Treatment20:InitWt	
-0.003365	-0.001082	

> anova(fm1Adg) # checking significance of terms

Analysis of Variance Table

	npair	Sum Sq	Mean Sq	F value
Treatment	4	5.7248	1.43119	28.9543
InitWt	1	0.5495	0.54953	11.1175
Treatment:InitWt	3	0.1381	0.04603	0.9312

> ## common slope model

> (fm2Adg <- lmer(adg ~ InitWt + Treatment + (1 | Block), AvgDailyGain))

Linear mixed model fit by REML ['lmerMod']

Formula: adg ~ InitWt + Treatment + (1 | Block)

Data: AvgDailyGain

REML criterion at convergence: 36.3373

Random effects:

Groups	Name	Std.Dev.
Block	(Intercept)	0.4908
Residual		0.2238

Number of obs: 32, groups: Block, 8

Fixed Effects:

(Intercept)	InitWt	Treatment0	Treatment10	Treatment20
0.80111	0.00278	-0.55207	-0.06857	-0.08813

> anova(fm2Adg)

Analysis of Variance Table

	npair	Sum Sq	Mean Sq	F value
InitWt	1	0.51455	0.51455	10.275
Treatment	3	1.52670	0.50890	10.162

> (fm3Adg <- lmer(adg ~ InitWt + Treatment - 1 + (1 | Block), AvgDailyGain))



```

Linear mixed model fit by REML ['lmerMod']
Formula: adg ~ InitWt + Treatment - 1 + (1 | Block)
  Data: AvgDailyGain
REML criterion at convergence: 36.3373
Random effects:
  Groups   Name                Std.Dev.
  Block    (Intercept) 0.4908
  Residual                    0.2238
Number of obs: 32, groups: Block, 8
Fixed Effects:
      InitWt   Treatment0   Treatment10   Treatment20   Treatment30
      0.00278      0.24903      0.73254      0.71298      0.80111

```

## B BIB

```

> print(xyplot(y ~ x | Block, BIB, groups = Treatment, type = c("g", "p"),
+          aspect = "xy", auto.key = list(points = TRUE, space = "right",
+          lines = FALSE)))

```

```

> ## compare with Output 5.7, p. 188

```

```

> (fm1BIB <- lmer(y ~ Treatment * x + (1|Block), BIB))

```

```

Linear mixed model fit by REML ['lmerMod']

```

```

Formula: y ~ Treatment * x + (1 | Block)

```

```

  Data: BIB

```

```

REML criterion at convergence: 104.8945

```

```

Random effects:

```

```

  Groups   Name                Std.Dev.
  Block    (Intercept) 4.272
  Residual                    1.096

```

```

Number of obs: 24, groups: Block, 8

```

```

Fixed Effects:

```

```

  (Intercept)   Treatment1   Treatment2   Treatment3           x
      22.36784      4.42949      -0.43737      6.27864      0.44255
Treatment1:x   Treatment2:x   Treatment3:x
      -0.22377      0.05338      -0.17918

```

```

> anova(fm1BIB) # strong evidence of different slopes

```

```

Analysis of Variance Table

```

```

      npar  Sum Sq Mean Sq  F value
Treatment    3  23.447   7.816   6.5110
x             1 136.809 136.809 113.9693
Treatment:x   3  18.427   6.142   5.1168

```

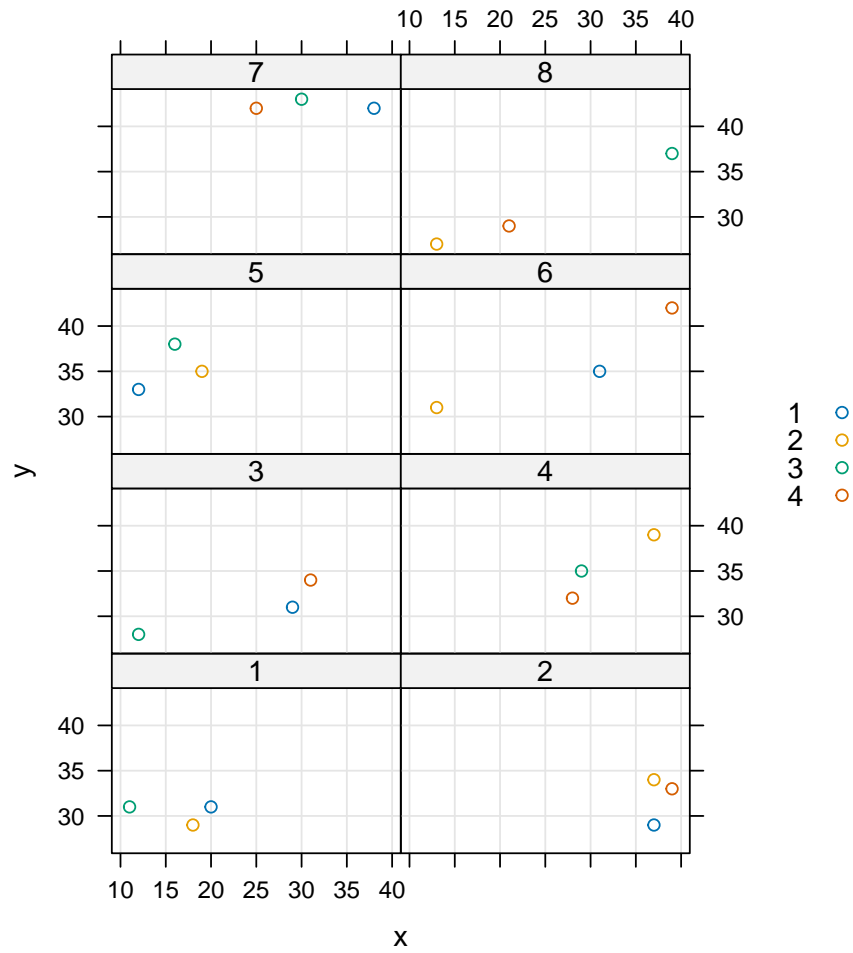


Figure 2: Balanced incomplete block design

```

> ## compare with Output 5.9, p. 193
> (fm2BIB <- lmer(y ~ Treatment + x:Grp + (1|Block), BIB))
Linear mixed model fit by REML ['lmerMod']
Formula: y ~ Treatment + x:Grp + (1 | Block)
Data: BIB
REML criterion at convergence: 99.177
Random effects:
Groups Name Std.Dev.
Block (Intercept) 4.304
Residual 1.019
Number of obs: 24, groups: Block, 8
Fixed Effects:
(Intercept) Treatment1 Treatment2 Treatment3 x:Grp13
20.9452 5.3414 1.1356 8.1810 0.2395
x:Grp24
0.4892
> anova(fm2BIB)
Analysis of Variance Table
npar Sum Sq Mean Sq F value
Treatment 3 23.424 7.808 7.5236
x:Grp 2 154.733 77.367 74.5471

```

## C Bond

```

> ## compare with output 1.1 on p. 6
> (fm1Bond <- lmer(pressure ~ Metal + (1|Ingot), Bond))
Linear mixed model fit by REML ['lmerMod']
Formula: pressure ~ Metal + (1 | Ingot)
Data: Bond
REML criterion at convergence: 107.7902
Random effects:
Groups Name Std.Dev.
Ingot (Intercept) 3.383
Residual 3.220
Number of obs: 21, groups: Ingot, 7
Fixed Effects:
(Intercept) Metalc Metali
71.1000 -0.9143 4.8000
> anova(fm1Bond)
Analysis of Variance Table
npar Sum Sq Mean Sq F value
Metal 2 131.9 65.95 6.3588

```

## D Cultivation

```
> str(Cultivation)
'data.frame':      24 obs. of  4 variables:
 $ Block: Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 2 2 2 2 ...
 $ Cult : Factor w/ 2 levels "a","b": 1 1 1 2 2 2 1 1 1 2 ...
 $ Inoc : Factor w/ 3 levels "con","dea","liv": 1 2 3 1 2 3 1 2 3 1 ...
 $ drywt: num  27.4 29.7 34.5 29.4 32.5 34.4 28.9 28.7 33.4 28.7 ...
- attr(*, "ginfo")=List of 7
 ..$ formula      :Class 'formula'  language drywt ~ 1 | Block/Cult
 .. .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
 ..$ order.groups:List of 2
 .. ..$ Block: logi TRUE
 .. ..$ Cult : logi TRUE
 ..$ FUN          :function (x)
 ..$ outer        : NULL
 ..$ inner        :List of 1
 .. ..$ Cult:Class 'formula'  language ~Inoc
 .. .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
 ..$ labels       :List of 1
 .. ..$ drywt: chr "Yield"
 ..$ units        : list()
> xtabs(~Block+Cult, Cultivation)
      Cult
Block a b
  1 3 3
  2 3 3
  3 3 3
  4 3 3
> (fmlCult <- lmer(drywt ~ Inoc * Cult + (1|Block) + (1|Cult), Cultivation))
Linear mixed model fit by REML ['lmerMod']
Formula: drywt ~ Inoc * Cult + (1 | Block) + (1 | Cult)
Data: Cultivation
REML criterion at convergence: 68.4874
Random effects:
Groups   Name             Std.Dev.
Block    (Intercept)  1.0988
Cult     (Intercept)  0.1606
Residual                    1.0938
Number of obs: 24, groups: Block, 4; Cult, 2
Fixed Effects:
```

```

      (Intercept)      Inoccon      Inocdea      Cultra  Inoccon:Cultra
      33.525         -5.500         -2.875         -0.375         0.250
Inocdea:Cultra
      -1.025
optimizer (nloptwrap) convergence code: 0 (OK) ; 0 optimizer warnings; 2 lme4
> anova(fm1Cult)
Analysis of Variance Table

      npar  Sum Sq Mean Sq F value
Inoc      2 118.176  59.088 49.3909
Cult      1   1.912   1.912  1.5981
Inoc:Cult  2   1.826   0.913  0.7631
> (fm2Cult <- lmer(drywt ~ Inoc + Cult + (1|Block) + (1|Cult), Cultivation))
Linear mixed model fit by REML ['lmerMod']
Formula: drywt ~ Inoc + Cult + (1 | Block) + (1 | Cult)
Data: Cultivation
REML criterion at convergence: 73.7535
Random effects:
  Groups   Name      Std.Dev.
Block     (Intercept) 1.1013
Cult      (Intercept) 0.1359
Residual                    1.0784
Number of obs: 24, groups: Block, 4; Cult, 2
Fixed Effects:
      (Intercept)      Inoccon      Inocdea      Cultra
      33.6542         -5.3750         -3.3875         -0.6333
optimizer (nloptwrap) convergence code: 0 (OK) ; 0 optimizer warnings; 2 lme4
> anova(fm2Cult)
Analysis of Variance Table

      npar  Sum Sq Mean Sq F value
Inoc      2 118.176  59.088 50.8069
Cult      1   2.021   2.021  1.7381
> (fm3Cult <- lmer(drywt ~ Inoc + (1|Block) + (1|Cult), Cultivation))
Linear mixed model fit by REML ['lmerMod']
Formula: drywt ~ Inoc + (1 | Block) + (1 | Cult)
Data: Cultivation
REML criterion at convergence: 75.6778
Random effects:
  Groups   Name      Std.Dev.
Block     (Intercept) 1.1013
Cult      (Intercept) 0.3219

```

```

Residual                1.0784
Number of obs: 24, groups: Block, 4; Cult, 2
Fixed Effects:
(Intercept)      Inoccon      Inocdea
                33.338       -5.375       -3.388
> anova(fm3Cult)
Analysis of Variance Table
      npar Sum Sq Mean Sq F value
Inoc   2 118.18  59.088  50.807

```

## E Demand

```

> ## compare to output 3.13, p. 132
> (fm1Demand <-
+ lmer(log(d) ~ log(y) + log(rd) + log(rt) + log(rs) + (1|State) + (1|Year),
+ Demand))
Linear mixed model fit by REML ['lmerMod']
Formula: log(d) ~ log(y) + log(rd) + log(rt) + log(rs) + (1 | State) +
(1 | Year)
Data: Demand
REML criterion at convergence: -240.1653
Random effects:
Groups   Name          Std.Dev.
Year     (Intercept)  0.01627
State    (Intercept)  0.17177
Residual                    0.03342
Number of obs: 77, groups: Year, 11; State, 7
Fixed Effects:
(Intercept)      log(y)      log(rd)      log(rt)      log(rs)
-1.28383        1.06978       -0.29533       0.03988       -0.32674

```

## F HR

```

> ## linear trend in time
> (fm1HR <- lmer(HR ~ Time * Drug + baseHR + (Time|Patient), HR))
Linear mixed model fit by REML ['lmerMod']
Formula: HR ~ Time * Drug + baseHR + (Time | Patient)
Data: HR
REML criterion at convergence: 767.607
Random effects:

```

```

Groups   Name          Std.Dev. Corr
Patient (Intercept)  7.787
          Time        6.147   -0.56
Residual                4.936
Number of obs: 120, groups: Patient, 24
Fixed Effects:
(Intercept)          Time          Druga          Drugb          baseHR
      33.9775         -3.1970          3.5992          7.0912          0.5434
Time:Druga   Time:Drugb
      -7.5013         -3.9894
> anova(fm1HR)
Analysis of Variance Table
              npar Sum Sq Mean Sq F value
Time          1  379.23   379.23 15.5671
Drug          2   92.88    46.44  1.9064
baseHR       1  533.28   533.28 21.8909
Time:Drug    2   72.12    36.06  1.4802
> ## remove interaction
> (fm3HR <- lmer(HR ~ Time + Drug + baseHR + (Time|Patient), HR))
Linear mixed model fit by REML ['lmerMod']
Formula: HR ~ Time + Drug + baseHR + (Time | Patient)
Data: HR
REML criterion at convergence: 779.8283
Random effects:
Groups   Name          Std.Dev. Corr
Patient (Intercept)  7.846
          Time        6.400   -0.57
Residual                4.936
Number of obs: 120, groups: Patient, 24
Fixed Effects:
(Intercept)          Time          Druga          Drugb          baseHR
      36.0481         -7.0273         -0.4529          4.9362          0.5434
> anova(fm3HR)
Analysis of Variance Table
              npar Sum Sq Mean Sq F value
Time          1  364.01   364.01 14.9428
Drug          2   92.88    46.44  1.9065
baseHR       1  533.22   533.22 21.8888
> ## remove Drug term
> (fm4HR <- lmer(HR ~ Time + baseHR + (Time|Patient), HR))

```

```

Linear mixed model fit by REML ['lmerMod']
Formula: HR ~ Time + baseHR + (Time | Patient)
  Data: HR
REML criterion at convergence: 791.1481
Random effects:
  Groups   Name                Std.Dev. Corr
Patient   (Intercept)  7.939
          Time          6.400   -0.55
Residual                    4.936
Number of obs: 120, groups: Patient, 24
Fixed Effects:
(Intercept)          Time          baseHR
    36.9318         -7.0273         0.5508
> anova(fm4HR)
Analysis of Variance Table
      npar Sum Sq Mean Sq F value
Time      1  364.02   364.02  14.943
baseHR    1  534.84   534.84  21.955

```

## G Mississippi

```

> ## compare with output 4.1, p. 142
> (fm1Miss <- lmer(y ~ 1 + (1 | influent), Mississippi))
Linear mixed model fit by REML ['lmerMod']
Formula: y ~ 1 + (1 | influent)
  Data: Mississippi
REML criterion at convergence: 252.3511
Random effects:
  Groups   Name                Std.Dev.
influent  (Intercept)  7.958
Residual                    6.531
Number of obs: 37, groups: influent, 6
Fixed Effects:
(Intercept)
    21.22
> ## compare with output 4.2, p. 143
> (fm1MLMiss <- lmer(y ~ 1 + (1 | influent), Mississippi, REML=FALSE))
Linear mixed model fit by maximum likelihood ['lmerMod']
Formula: y ~ 1 + (1 | influent)
  Data: Mississippi

```



```

          AIC          BIC      logLik  deviance  df.resid
262.5570  267.3898 -128.2785  256.5570      34
Random effects:
  Groups   Name      Std.Dev.
influent (Intercept) 7.159
Residual                6.534
Number of obs: 37, groups:  influenza, 6
Fixed Effects:
(Intercept)
      21.22
> ranef(fm1MLMiss)          # BLUP's of random effects on p. 144
$influent
  (Intercept)
1    0.3097833
2   -6.5772282
3   -3.7862751
4    2.8826713
5   -5.8435214
6   13.0145701

with conditional variances for \influent"
> ranef(fm1Miss)          # BLUP's of random effects on p. 142
$influent
  (Intercept)
1    0.309286
2   -6.719332
3   -3.897945
4    2.946104
5   -6.012984
6   13.374871

with conditional variances for \influent"
> VarCorr(fm1Miss)          # compare to output 4.7, p. 148
  Groups   Name      Std.Dev.
influent (Intercept) 7.9576
Residual                6.5313
> ## compare to output 4.8 and 4.9, pp. 150-152
> (fm2Miss <- lmer(y ~ Type + (1 | influenza), Mississippi))

```

```

Linear mixed model fit by REML ['lmerMod']
Formula: y ~ Type + (1 | influent)
  Data: Mississippi
REML criterion at convergence: 234.5246
Random effects:
  Groups   Name                Std.Dev.
influent (Intercept) 3.869
Residual                    6.520
Number of obs: 37, groups: influent, 6
Fixed Effects:
(Intercept)          Type1          Type2
          36.40          -20.80          -16.46
> anova(fm2Miss)
Analysis of Variance Table
      npar Sum Sq Mean Sq F value
Type    2 541.76  270.88  6.3716

```

## H Multilocation

```

> str(Multilocation)
'data.frame':   108 obs. of  7 variables:
 $ obs      : num  3 4 6 7 9 10 12 16 19 20 ...
 $ Location: Factor w/ 9 levels "A","B","C","D",...: 1 1 1 1 1 1 1 1 1 1 ...
 $ Block    : Factor w/ 3 levels "1","2","3": 1 1 1 1 2 2 2 2 3 3 ...
 $ Trt      : Factor w/ 4 levels "1","2","3","4": 3 4 2 1 2 1 3 4 1 2 ...
 $ Adj      : num  3.16 3.12 3.16 3.25 2.71 ...
 $ Fe       : num  7.1 6.68 6.83 6.53 8.25 ...
 $ Grp      : Factor w/ 27 levels "A/1","A/2","A/3",...: 1 1 1 1 2 2 2 2 3 3 ...
- attr(*, "ginfo")=List of 7
 ..$ formula      :Class 'formula' language Adj ~ 1 | Location/Block
 .. .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
 ..$ order.groups:List of 2
 .. ..$ Location: logi TRUE
 .. ..$ Block    : logi TRUE
 ..$ FUN          :function (x)
 ..$ outer        : NULL
 ..$ inner        :List of 1
 .. ..$ Block:Class 'formula' language ~Trt
 .. .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
 ..$ labels       :List of 1

```

```

.. ..$ Adj: chr "Adjusted yield"
..$ units      : list()
> ### Create a Block %in% Location factor
> Multilocation$Grp <- with(Multilocation, Block:Location)
> (fm1Mult <- lmer(Adj ~ Location * Trt + (1|Grp), Multilocation))
Linear mixed model fit by REML ['lmerMod']
Formula: Adj ~ Location * Trt + (1 | Grp)
Data: Multilocation
REML criterion at convergence: 10.6462
Random effects:
Groups   Name                Std.Dev.
Grp      (Intercept) 0.07496
Residual                    0.18595
Number of obs: 108, groups: Grp, 27
Fixed Effects:
(Intercept)      LocationA      LocationB      LocationC
2.35923          0.64930          0.06643          0.54533
LocationD      LocationE      LocationF      LocationG
0.37413          0.55000          0.99810          0.36057
LocationH      Trt1          Trt2          Trt3
1.01403          0.22720         -0.00140          0.42323
LocationA:Trt1 LocationB:Trt1 LocationC:Trt1 LocationD:Trt1
-0.18853         -0.27523         -0.04000         -0.53513
LocationE:Trt1 LocationF:Trt1 LocationG:Trt1 LocationH:Trt1
-0.26297         -0.27153          0.20323         -0.14953
LocationA:Trt2 LocationB:Trt2 LocationC:Trt2 LocationD:Trt2
-0.09347         -0.32273          0.08960         -0.29693
LocationE:Trt2 LocationF:Trt2 LocationG:Trt2 LocationH:Trt2
-0.30693         -0.30993         -0.10860         -0.33060
LocationA:Trt3 LocationB:Trt3 LocationC:Trt3 LocationD:Trt3
-0.40247         -0.56550         -0.12247         -0.54840
LocationE:Trt3 LocationF:Trt3 LocationG:Trt3 LocationH:Trt3
-0.32863         -0.46257         -0.25297         -0.37203
> anova(fm1Mult)
Analysis of Variance Table
              npar Sum Sq Mean Sq F value
Location      8  6.9475  0.86843  25.1147
Trt           3  1.2217  0.40725  11.7774
Location:Trt 24  0.9966  0.04152   1.2008
> (fm2Mult <- lmer(Adj ~ Location + Trt + (1|Grp), Multilocation))

```

```

Linear mixed model fit by REML ['lmerMod']
Formula: Adj ~ Location + Trt + (1 | Grp)
  Data: Multilocation
REML criterion at convergence: -6.0011
Random effects:
  Groups   Name                Std.Dev.
  Grp      (Intercept) 0.07131
  Residual                                0.19161
Number of obs: 108, groups:  Grp, 27
Fixed Effects:
(Intercept)      LocationA      LocationB      LocationC      LocationD
      2.53296      0.47818      -0.22443      0.52712      0.02902
  LocationE      LocationF      LocationG      LocationH      Trt1
      0.32537      0.73709      0.32098      0.80099      0.05834
      Trt2      Trt3
      -0.18802      0.08379
> (fm3Mult <- lmer(Adj ~ Location + (1|Grp), Multilocation))
Linear mixed model fit by REML ['lmerMod']
Formula: Adj ~ Location + (1 | Grp)
  Data: Multilocation
REML criterion at convergence: 9.8205
Random effects:
  Groups   Name                Std.Dev.
  Grp      (Intercept) 0.04067
  Residual                                0.22459
Number of obs: 108, groups:  Grp, 27
Fixed Effects:
(Intercept)      LocationA      LocationB      LocationC      LocationD
      2.52149      0.47818      -0.22443      0.52712      0.02902
  LocationE      LocationF      LocationG      LocationH
      0.32537      0.73709      0.32098      0.80099
> (fm4Mult <- lmer(Adj ~ Trt + (1|Grp), Multilocation))
Linear mixed model fit by REML ['lmerMod']
Formula: Adj ~ Trt + (1 | Grp)
  Data: Multilocation
REML criterion at convergence: 31.5057
Random effects:
  Groups   Name                Std.Dev.
  Grp      (Intercept) 0.3331
  Residual                                0.1916

```

```

Number of obs: 108, groups: Grp, 27
Fixed Effects:
(Intercept)      Trt1      Trt2      Trt3
      2.86567      0.05834     -0.18802      0.08379
> (fm5Mult <- lmer(Adj ~ 1 + (1|Grp), Multilocation))
Linear mixed model fit by REML ['lmerMod']
Formula: Adj ~ 1 + (1 | Grp)
Data: Multilocation
REML criterion at convergence: 47.3273
Random effects:
Groups Name Std.Dev.
Grp (Intercept) 0.3279
Residual 0.2246
Number of obs: 108, groups: Grp, 27
Fixed Effects:
(Intercept)
      2.854
> anova(fm2Mult)
Analysis of Variance Table
      npar Sum Sq Mean Sq F value
Location      8 7.3768 0.92210 25.115
Trt           3 1.2217 0.40725 11.092
> fm2MultR <- lmer(Adj ~ Trt + (Trt - 1|Location) + (1|Block), Multilocation,
+ verbose = TRUE)
iteration: 1
      f(x) = 62.845600
iteration: 2
      f(x) = 60.463797
iteration: 3
      f(x) = 53.469241
iteration: 4
      f(x) = 51.917416
iteration: 5
      f(x) = 53.713467
iteration: 6
      f(x) = 62.710254
iteration: 7
      f(x) = 51.828252
iteration: 8
      f(x) = 53.008611

```

```
iteration: 9
      f(x) = 61.842744
iteration: 10
      f(x) = 53.489669
iteration: 11
      f(x) = 61.977590
iteration: 12
      f(x) = 65.037152
iteration: 13
      f(x) = 84.552209
iteration: 14
      f(x) = 83.407297
iteration: 15
      f(x) = 84.212059
iteration: 16
      f(x) = 82.929486
iteration: 17
      f(x) = 78.420892
iteration: 18
      f(x) = 81.586200
iteration: 19
      f(x) = 80.745840
iteration: 20
      f(x) = 80.847450
iteration: 21
      f(x) = 81.442598
iteration: 22
      f(x) = 80.475277
iteration: 23
      f(x) = 58.139221
iteration: 24
      f(x) = 32.909702
iteration: 25
      f(x) = 27.744930
iteration: 26
      f(x) = 25.344261
iteration: 27
      f(x) = 25.515267
iteration: 28
      f(x) = 23.786171
```

```
iteration: 29
      f(x) = 22.862422
iteration: 30
      f(x) = 23.188470
iteration: 31
      f(x) = 22.989693
iteration: 32
      f(x) = 22.582913
iteration: 33
      f(x) = 22.886783
iteration: 34
      f(x) = 22.908443
iteration: 35
      f(x) = 22.374106
iteration: 36
      f(x) = 22.957737
iteration: 37
      f(x) = 22.388049
iteration: 38
      f(x) = 21.524610
iteration: 39
      f(x) = 20.049672
iteration: 40
      f(x) = 18.500236
iteration: 41
      f(x) = 18.001011
iteration: 42
      f(x) = 18.330995
iteration: 43
      f(x) = 19.193142
iteration: 44
      f(x) = 18.079719
iteration: 45
      f(x) = 18.225494
iteration: 46
      f(x) = 17.211372
iteration: 47
      f(x) = 15.921887
iteration: 48
      f(x) = 15.141220
```

```
iteration: 49
      f(x) = 15.133217
iteration: 50
      f(x) = 16.318467
iteration: 51
      f(x) = 14.330844
iteration: 52
      f(x) = 13.088787
iteration: 53
      f(x) = 18.725331
iteration: 54
      f(x) = 14.092576
iteration: 55
      f(x) = 10.184134
iteration: 56
      f(x) = 9.694098
iteration: 57
      f(x) = 12.079162
iteration: 58
      f(x) = 9.833605
iteration: 59
      f(x) = 9.270919
iteration: 60
      f(x) = 8.673704
iteration: 61
      f(x) = 9.050488
iteration: 62
      f(x) = 8.559737
iteration: 63
      f(x) = 8.540362
iteration: 64
      f(x) = 8.752601
iteration: 65
      f(x) = 8.255258
iteration: 66
      f(x) = 8.682453
iteration: 67
      f(x) = 8.461774
iteration: 68
      f(x) = 7.869992
```



```
iteration: 69
      f(x) = 8.716852
iteration: 70
      f(x) = 8.316755
iteration: 71
      f(x) = 6.563278
iteration: 72
      f(x) = 5.518809
iteration: 73
      f(x) = 5.059824
iteration: 74
      f(x) = 3.815827
iteration: 75
      f(x) = 3.904434
iteration: 76
      f(x) = 4.318221
iteration: 77
      f(x) = 3.728094
iteration: 78
      f(x) = 3.870752
iteration: 79
      f(x) = 3.850116
iteration: 80
      f(x) = 3.523378
iteration: 81
      f(x) = 3.777261
iteration: 82
      f(x) = 3.057532
iteration: 83
      f(x) = 2.925564
iteration: 84
      f(x) = 2.659732
iteration: 85
      f(x) = 2.466434
iteration: 86
      f(x) = 2.338609
iteration: 87
      f(x) = 2.248804
iteration: 88
      f(x) = 2.152620
```

```
iteration: 89
      f(x) = 2.130636
iteration: 90
      f(x) = 2.147609
iteration: 91
      f(x) = 2.322373
iteration: 92
      f(x) = 2.048578
iteration: 93
      f(x) = 2.018397
iteration: 94
      f(x) = 2.078101
iteration: 95
      f(x) = 2.048181
iteration: 96
      f(x) = 2.070543
iteration: 97
      f(x) = 2.079374
iteration: 98
      f(x) = 2.079822
iteration: 99
      f(x) = 1.990186
iteration: 100
      f(x) = 1.916379
iteration: 101
      f(x) = 1.977139
iteration: 102
      f(x) = 1.957163
iteration: 103
      f(x) = 1.947564
iteration: 104
      f(x) = 1.953546
iteration: 105
      f(x) = 1.882586
iteration: 106
      f(x) = 1.873437
iteration: 107
      f(x) = 1.854022
iteration: 108
      f(x) = 1.833124
```

iteration: 109  
f(x) = 1.851868  
iteration: 110  
f(x) = 1.938274  
iteration: 111  
f(x) = 1.841808  
iteration: 112  
f(x) = 1.834307  
iteration: 113  
f(x) = 1.841994  
iteration: 114  
f(x) = 1.835919  
iteration: 115  
f(x) = 1.817128  
iteration: 116  
f(x) = 1.783542  
iteration: 117  
f(x) = 1.743145  
iteration: 118  
f(x) = 1.702998  
iteration: 119  
f(x) = 1.751397  
iteration: 120  
f(x) = 1.672946  
iteration: 121  
f(x) = 1.703653  
iteration: 122  
f(x) = 1.683640  
iteration: 123  
f(x) = 1.621276  
iteration: 124  
f(x) = 1.597704  
iteration: 125  
f(x) = 1.619830  
iteration: 126  
f(x) = 1.596416  
iteration: 127  
f(x) = 1.574240  
iteration: 128  
f(x) = 1.583606

iteration: 129  
f(x) = 1.569989  
iteration: 130  
f(x) = 1.589155  
iteration: 131  
f(x) = 1.568331  
iteration: 132  
f(x) = 1.562786  
iteration: 133  
f(x) = 1.555114  
iteration: 134  
f(x) = 1.548216  
iteration: 135  
f(x) = 1.556342  
iteration: 136  
f(x) = 1.549113  
iteration: 137  
f(x) = 1.543484  
iteration: 138  
f(x) = 1.541380  
iteration: 139  
f(x) = 1.541076  
iteration: 140  
f(x) = 1.539990  
iteration: 141  
f(x) = 1.540902  
iteration: 142  
f(x) = 1.535965  
iteration: 143  
f(x) = 1.533789  
iteration: 144  
f(x) = 1.535703  
iteration: 145  
f(x) = 1.531284  
iteration: 146  
f(x) = 1.529555  
iteration: 147  
f(x) = 1.530255  
iteration: 148  
f(x) = 1.530102

```
iteration: 149
      f(x) = 1.527541
iteration: 150
      f(x) = 1.528067
iteration: 151
      f(x) = 1.528401
iteration: 152
      f(x) = 1.528611
iteration: 153
      f(x) = 1.525588
iteration: 154
      f(x) = 1.520881
iteration: 155
      f(x) = 1.514761
iteration: 156
      f(x) = 1.507301
iteration: 157
      f(x) = 1.506880
iteration: 158
      f(x) = 1.507934
iteration: 159
      f(x) = 1.517716
iteration: 160
      f(x) = 1.502957
iteration: 161
      f(x) = 1.500955
iteration: 162
      f(x) = 1.496610
iteration: 163
      f(x) = 1.488485
iteration: 164
      f(x) = 1.492998
iteration: 165
      f(x) = 1.481736
iteration: 166
      f(x) = 1.474592
iteration: 167
      f(x) = 1.472433
iteration: 168
      f(x) = 1.473707
```

iteration: 169  
f(x) = 1.470600  
iteration: 170  
f(x) = 1.468352  
iteration: 171  
f(x) = 1.466728  
iteration: 172  
f(x) = 1.461302  
iteration: 173  
f(x) = 1.457605  
iteration: 174  
f(x) = 1.453063  
iteration: 175  
f(x) = 1.453212  
iteration: 176  
f(x) = 1.474920  
iteration: 177  
f(x) = 1.452026  
iteration: 178  
f(x) = 1.450846  
iteration: 179  
f(x) = 1.450411  
iteration: 180  
f(x) = 1.447881  
iteration: 181  
f(x) = 1.447359  
iteration: 182  
f(x) = 1.446797  
iteration: 183  
f(x) = 1.445069  
iteration: 184  
f(x) = 1.439161  
iteration: 185  
f(x) = 1.439199  
iteration: 186  
f(x) = 1.446981  
iteration: 187  
f(x) = 1.437353  
iteration: 188  
f(x) = 1.438391

```
iteration: 189
      f(x) = 1.443217
iteration: 190
      f(x) = 1.435385
iteration: 191
      f(x) = 1.437797
iteration: 192
      f(x) = 1.445218
iteration: 193
      f(x) = 1.432786
iteration: 194
      f(x) = 1.431487
iteration: 195
      f(x) = 1.432014
iteration: 196
      f(x) = 1.438837
iteration: 197
      f(x) = 1.433773
iteration: 198
      f(x) = 1.443164
iteration: 199
      f(x) = 1.429641
iteration: 200
      f(x) = 1.431093
iteration: 201
      f(x) = 1.430602
iteration: 202
      f(x) = 1.429214
iteration: 203
      f(x) = 1.429360
iteration: 204
      f(x) = 1.428242
iteration: 205
      f(x) = 1.428818
iteration: 206
      f(x) = 1.429682
iteration: 207
      f(x) = 1.427753
iteration: 208
      f(x) = 1.427824
```

iteration: 209  
f(x) = 1.427172  
iteration: 210  
f(x) = 1.426241  
iteration: 211  
f(x) = 1.425394  
iteration: 212  
f(x) = 1.425019  
iteration: 213  
f(x) = 1.425752  
iteration: 214  
f(x) = 1.431352  
iteration: 215  
f(x) = 1.424836  
iteration: 216  
f(x) = 1.424795  
iteration: 217  
f(x) = 1.429056  
iteration: 218  
f(x) = 1.424965  
iteration: 219  
f(x) = 1.429383  
iteration: 220  
f(x) = 1.422907  
iteration: 221  
f(x) = 1.422640  
iteration: 222  
f(x) = 1.425139  
iteration: 223  
f(x) = 1.424562  
iteration: 224  
f(x) = 1.423515  
iteration: 225  
f(x) = 1.424887  
iteration: 226  
f(x) = 1.422926  
iteration: 227  
f(x) = 1.424799  
iteration: 228  
f(x) = 1.422240



```
iteration: 229
      f(x) = 1.424293
iteration: 230
      f(x) = 1.424857
iteration: 231
      f(x) = 1.422348
iteration: 232
      f(x) = 1.423076
iteration: 233
      f(x) = 1.421268
iteration: 234
      f(x) = 1.420036
iteration: 235
      f(x) = 1.425189
iteration: 236
      f(x) = 1.419935
iteration: 237
      f(x) = 1.419314
iteration: 238
      f(x) = 1.418114
iteration: 239
      f(x) = 1.419019
iteration: 240
      f(x) = 1.417896
iteration: 241
      f(x) = 1.418284
iteration: 242
      f(x) = 1.419151
iteration: 243
      f(x) = 1.417242
iteration: 244
      f(x) = 1.417257
iteration: 245
      f(x) = 1.417762
iteration: 246
      f(x) = 1.416699
iteration: 247
      f(x) = 1.415988
iteration: 248
      f(x) = 1.415043
```

iteration: 249  
f(x) = 1.414162  
iteration: 250  
f(x) = 1.413969  
iteration: 251  
f(x) = 1.414165  
iteration: 252  
f(x) = 1.415707  
iteration: 253  
f(x) = 1.414059  
iteration: 254  
f(x) = 1.414749  
iteration: 255  
f(x) = 1.413905  
iteration: 256  
f(x) = 1.414115  
iteration: 257  
f(x) = 1.413709  
iteration: 258  
f(x) = 1.413285  
iteration: 259  
f(x) = 1.412806  
iteration: 260  
f(x) = 1.412105  
iteration: 261  
f(x) = 1.411196  
iteration: 262  
f(x) = 1.409932  
iteration: 263  
f(x) = 1.411190  
iteration: 264  
f(x) = 1.411802  
iteration: 265  
f(x) = 1.410035  
iteration: 266  
f(x) = 1.412616  
iteration: 267  
f(x) = 1.409214  
iteration: 268  
f(x) = 1.408619

iteration: 269  
f(x) = 1.408777  
iteration: 270  
f(x) = 1.415514  
iteration: 271  
f(x) = 1.408535  
iteration: 272  
f(x) = 1.408540  
iteration: 273  
f(x) = 1.408593  
iteration: 274  
f(x) = 1.408512  
iteration: 275  
f(x) = 1.408657  
iteration: 276  
f(x) = 1.408544  
iteration: 277  
f(x) = 1.408399  
iteration: 278  
f(x) = 1.408205  
iteration: 279  
f(x) = 1.408049  
iteration: 280  
f(x) = 1.408007  
iteration: 281  
f(x) = 1.407895  
iteration: 282  
f(x) = 1.407748  
iteration: 283  
f(x) = 1.407711  
iteration: 284  
f(x) = 1.407697  
iteration: 285  
f(x) = 1.407667  
iteration: 286  
f(x) = 1.407666  
iteration: 287  
f(x) = 1.407657  
iteration: 288  
f(x) = 1.407685

iteration: 289  
f(x) = 1.407654  
iteration: 290  
f(x) = 1.407671  
iteration: 291  
f(x) = 1.407649  
iteration: 292  
f(x) = 1.407635  
iteration: 293  
f(x) = 1.407626  
iteration: 294  
f(x) = 1.407720  
iteration: 295  
f(x) = 1.407565  
iteration: 296  
f(x) = 1.407493  
iteration: 297  
f(x) = 1.407413  
iteration: 298  
f(x) = 1.407409  
iteration: 299  
f(x) = 1.407428  
iteration: 300  
f(x) = 1.407369  
iteration: 301  
f(x) = 1.407317  
iteration: 302  
f(x) = 1.407338  
iteration: 303  
f(x) = 1.407349  
iteration: 304  
f(x) = 1.407278  
iteration: 305  
f(x) = 1.407272  
iteration: 306  
f(x) = 1.407264  
iteration: 307  
f(x) = 1.407258  
iteration: 308  
f(x) = 1.407252

```
iteration: 309
      f(x) = 1.407256
iteration: 310
      f(x) = 1.407252
iteration: 311
      f(x) = 1.407253
iteration: 312
      f(x) = 1.407302
iteration: 313
      f(x) = 1.407229
iteration: 314
      f(x) = 1.407219
iteration: 315
      f(x) = 1.407221
iteration: 316
      f(x) = 1.407267
iteration: 317
      f(x) = 1.407215
iteration: 318
      f(x) = 1.407214
iteration: 319
      f(x) = 1.407213
iteration: 320
      f(x) = 1.407216
iteration: 321
      f(x) = 1.407227
iteration: 322
      f(x) = 1.407210
iteration: 323
      f(x) = 1.407209
iteration: 324
      f(x) = 1.407230
iteration: 325
      f(x) = 1.407211
iteration: 326
      f(x) = 1.407215
iteration: 327
      f(x) = 1.407208
iteration: 328
      f(x) = 1.407222
```

iteration: 329  
f(x) = 1.407229  
iteration: 330  
f(x) = 1.407209  
iteration: 331  
f(x) = 1.407213  
iteration: 332  
f(x) = 1.407207  
iteration: 333  
f(x) = 1.407206  
iteration: 334  
f(x) = 1.407206  
iteration: 335  
f(x) = 1.407216  
iteration: 336  
f(x) = 1.407206  
iteration: 337  
f(x) = 1.407206  
iteration: 338  
f(x) = 1.407206  
iteration: 339  
f(x) = 1.407206  
iteration: 340  
f(x) = 1.407206  
iteration: 341  
f(x) = 1.407206  
iteration: 342  
f(x) = 1.407206  
iteration: 343  
f(x) = 1.407205  
iteration: 344  
f(x) = 1.407205  
iteration: 345  
f(x) = 1.407205  
iteration: 346  
f(x) = 1.407204  
iteration: 347  
f(x) = 1.407206  
iteration: 348  
f(x) = 1.407207

iteration: 349  
f(x) = 1.407205  
iteration: 350  
f(x) = 1.407205  
iteration: 351  
f(x) = 1.407204  
iteration: 352  
f(x) = 1.407204  
iteration: 353  
f(x) = 1.407204  
iteration: 354  
f(x) = 1.407204  
iteration: 355  
f(x) = 1.407204  
iteration: 356  
f(x) = 1.407204  
iteration: 357  
f(x) = 1.407204  
iteration: 358  
f(x) = 1.407205  
iteration: 359  
f(x) = 1.407204  
iteration: 360  
f(x) = 1.407204  
iteration: 361  
f(x) = 1.407204  
iteration: 362  
f(x) = 1.407204  
iteration: 363  
f(x) = 1.407204  
iteration: 364  
f(x) = 1.407204  
iteration: 365  
f(x) = 1.407204  
iteration: 366  
f(x) = 1.407204  
iteration: 367  
f(x) = 1.407204  
iteration: 368  
f(x) = 1.407204

```

> ## non convergence in 10000 evaluations
> fm2MultR
Linear mixed model fit by REML ['lmerMod']
Formula: Adj ~ Trt + (Trt - 1 | Location) + (1 | Block)
Data: Multilocation
REML criterion at convergence: 1.4072
Random effects:
Groups   Name             Std.Dev. Corr
Location Trt1          0.3686
          Trt2          0.3271  0.99
          Trt3          0.3451  1.00 1.00
          Trt4          0.3378  0.93 0.97 0.95
Block    (Intercept) 0.0000
Residual                0.1944
Number of obs: 108, groups: Location, 9; Block, 3
Fixed Effects:
(Intercept)          Trt1          Trt2          Trt3
      2.86567      0.05834     -0.18802      0.08379
optimizer (nloptwrap) convergence code: 0 (OK) ; 0 optimizer warnings; 1 lme4

```

## I PBIB

```

> str(PBIB)
'data.frame':      60 obs. of  3 variables:
 $ response : num  2.4 2.5 2.6 2 2.7 2.8 2.4 2.7 2.6 2.8 ...
 $ Treatment: Factor w/ 15 levels "1","10","11",...: 7 15 1 5 11 13 14 1 2 1 ...
 $ Block    : Factor w/ 15 levels "1","10","11",...: 1 1 1 1 8 8 8 8 9 9 ...
- attr(*, "ginfo")=List of 7
 ..$ formula      :Class 'formula' language response ~ Treatment | Block
 .. .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
 ..$ order.groups: logi TRUE
 ..$ FUN          :function (x)
 ..$ outer       : NULL
 ..$ inner       : NULL
 ..$ labels      : list()
 ..$ units       : list()
> ## compare with output 1.7 pp. 24-25
> (fm1PBIB <- lmer(response ~ Treatment + (1 | Block), PBIB))
Linear mixed model fit by REML ['lmerMod']
Formula: response ~ Treatment + (1 | Block)

```



```

Data: PBIB
REML criterion at convergence: 51.9849
Random effects:
  Groups   Name      Std.Dev.
Block     (Intercept) 0.2157
Residual                          0.2925
Number of obs: 60, groups: Block, 15
Fixed Effects:
(Intercept)  Treatment1  Treatment10  Treatment11  Treatment12
  2.891311    -0.073789    -0.400249     0.007388     0.161510
Treatment13  Treatment14  Treatment15  Treatment2    Treatment3
 -0.273542    -0.400000    -0.032078    -0.485996    -0.436368
Treatment4   Treatment5   Treatment6   Treatment7    Treatment8
 -0.107481    -0.086413     0.019383    -0.102326    -0.109706

```

## J SIMS

```

> str(SIMS)
'data.frame':      3691 obs. of  3 variables:
 $ Pretot: num  29 38 31 31 29 23 23 33 30 32 ...
 $ Gain   : num   2  0  6  6  5  9  7  2  1  3 ...
 $ Class  : Factor w/ 190 levels "1","10","100",...: 1 1 1 1 1 1 1 1 1 1 ...
- attr(*, "ginfo")=List of 7
 ..$ formula      :Class 'formula' language Gain ~ Pretot | Class
 .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
 ..$ order.groups: logi TRUE
 ..$ FUN          :function (x)
 ..$ outer        : NULL
 ..$ inner        : NULL
 ..$ labels       :List of 2
 .. ..$ Pretot: chr "Sum of pre-test core item scores"
 .. ..$ Gain   : chr "Gain in mathematics achievement score"
 ..$ units       : list()
> ## compare to output 7.4, p. 262
> (fm1SIMS <- lmer(Gain ~ Pretot + (Pretot | Class), SIMS))
Linear mixed model fit by REML ['lmerMod']
Formula: Gain ~ Pretot + (Pretot | Class)
Data: SIMS
REML criterion at convergence: 22380.57
Random effects:

```

Groups	Name	Std.Dev.	Corr
Class	(Intercept)	3.80623	
	Pretot	0.09592	-0.64
Residual		4.71554	

Number of obs: 3691, groups: Class, 190

Fixed Effects:

(Intercept)	Pretot
7.059	-0.186

optimizer (nloptwrap) convergence code: 0 (OK) ; 0 optimizer warnings; 1 lme4