Harvard Department of Government 2003
Faraway Chapter 10, Random Effects

JEFF GILL
Visiting Professor, Fall 2024

Vocabulary Fun

From Andy Gelman's blog... All the names for hierarchical and multilevel modeling, Posted by Bob Carpenter on 18 September 2019, 3:00 pm

- hierarchical model: a multilevel model with a single nested hierarchy (note my nod to Quine's "Two Dogmas" with circular references)
- multilevel model: a hierarchical model with multiple non-nested hierarchies
- random effects model: Item-level parameters are often called "random effects"; reading all the ways the term is used on the Wikipedia page on random effects illustrates why Andrew dislikes the term so much (see also here; both links added by Andrew)—it means many different things to different communities.
- mixed effects model: that's a random effects model with some regular "fixed effect" regression thrown in; this is where lme4 is named after linear mixed effects and NONMEM after nonlinear mixed effects models.
- empirical Bayes: Near and dear to Andrew's heart, because regular Bayes just isn't empirical enough. I jest—it's because "empirical Bayes" means using maximum marginal likelihood to estimate priors from data (just like lme4 does).
- regularized/penalized/shrunk regression: common approach in machine learning where held out data is used to "learn" the regularization parameters, which are typically framed as shrinkage or regularization scales in penalty terms rather than as priors
- automatic relevance determination (ARD): Radford Neal's term in his thesis on Gaussian processes and now widely adopted in the GP literature
- domain adaptation: This one's common in the machine-learning literature; I think it came from Hal Daumé III's paper, "Frustratingly easy domain adaptation" in which he rediscovered the technique; he also calls logistic regression a "maximum entropy classifier," like many people in natural language processing (and physics)
- variance components model: I just learned this one on the Wikipedia page on random effects models
- cross-sectional (time-series) model: apparently a thing in econometrics
- nested data model, split-plot design, random coefficient: The Wikipedia page on multilevel models listed all these.
- iterated nested Laplace approximation (INLA), expectation maximization (EM): Popular algorithmic approaches that get confused with the modeling technique.

Vocabulary Overview

- For the data matrices, X_i for individual i in cluster j, and $\mathbf{Z}_j$ for cluster j, there are five canonical models that we will look at:

"Completely Pooled"	$y_i = \beta_0 + \beta_1 \mathbf{X}_i + \gamma \mathbf{Z} + e_i$
"Fixed Effect"	$y_i = \beta_{ij0} + \beta_1 \mathbf{X}_i + e_i$
"Random Effect"	$y_i = \beta_{ij0} + \beta_1 \mathbf{X}_i + \gamma \mathbf{Z}_j + e_i$
"Random Intercept and Random Slope"	$y_i = \beta_{ij0} + \beta_{ij1} \mathbf{X}_i + e_i$
"Completely Unpooled'	$y_{ij} = \beta_{j0} + (\beta_{j1} \mathbf{X}_{ij} + \gamma \mathbf{Z}_j) + e_{ij}$

- This is produced by replacing the previous γ coefficient names with common regression-style language.
- "Fixed" and "random" can differ in definition by literature (Kreft and De Leeuw 1988, Section 1.3.3, Gelman 2005), and better notation is "random intercepts" for "fixed effect," and "varying-intercept, varying-slope" for "random intercept and random slope."
- Best to conceptualize these specifications as members of a larger multilevel family where indices are *turned-on* or *turned-off* systematically depending on the hierarchical purpose.

Advantages of Multilevel Models

- Removes the restriction that the estimated coefficients are constant across individual cases by specifying levels of additional effects.
- Provides a notationally efficient way to organize groups in the model.
- Accounting for individual versus group-level variation.
- Modeling variation among individual-level regression coefficients.
- Estimating regression coefficients for groups of interest.
- Gets the standard errors right.

Features of Multilevel Models

- Each level of the model is its *own* regression, with its own assumptions about: functional form, linearity, independence, variance, distribution of errors, etc.
- Models are usually "mixed," meaning some coefficients are *modeled* and some are *unmodeld*.
- Multilevel models are highly symbiotic with Bayesian specifications because the focus in both cases is on making reasonable distributional assumptions.
- These approaches are generally more demanding of statistical estimation process (software) to produce results.

Linear Model Illustration

- Start with a standard linear model specification indexed by subjects and a first level of grouping, the *context* level.
- Now use a single explanatory variable that has the form:

$$y_i = \beta_{ij0} + \beta_{ij1} X_i + \epsilon_i.$$

- Suppose we have group-level explanatory variables, $Z_{j.}$, in that their effect is measured at the aggregated rather than at the individual level.
- Now add a second level to the model that explicitly nests effects within groups and index these groups $j = 1$ to J:

$$\begin{aligned} \beta_{ij} &= \gamma_{00} + \gamma_{10} Z_{j0} + u_{j0} \\ \beta_{ij} &= \gamma_{01} + \gamma_{11} Z_{j1} + u_{j1}, \end{aligned}$$

Linear Model Illustration

- The two-level model is produced by inserting the context level specifications into the original linear expression for the outcome variable of interest:

$$y_i = \gamma_{00} + \gamma_{01}X_i + \gamma_{10}Z_{j0} + \gamma_{11}X_iZ_{j1} + u_{j1}X_i + u_{j0} + \epsilon_i.$$

- This equation shows that the composite error structure, $u_{j1}X_i + u_{j0} + \epsilon_i$, is now clearly heteroscedastic since it is conditioned on levels of the explanatory variable, causing additional estimation complexity.
- Notice that there is an "automatic" interaction component: $\gamma_{11}X_iZ_{j1}$.
- Now we are going model *distributions* for β_{j0}, and β_{j1}.
- Thus these are called "random effects."

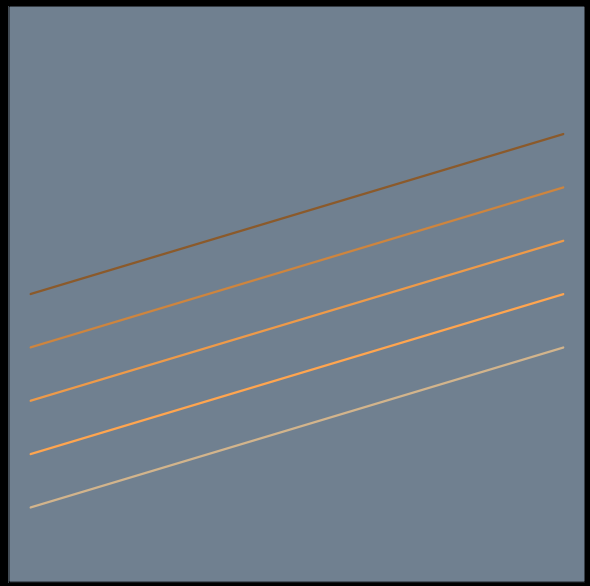
Varying-Intercept Models

- In the linear context, V-I models include indicators for groups who have the same slope but start at different points along the y-axis.
- Formally for $i = 1, \ldots, n$:

$$y_i = \beta_{ij} + \beta X_i + \epsilon_i$$

  where ij indicates that the ith case gets intercept j for the jth group. Some authors simply index this term as α_j since it is apparent that we are describing the ith case.
- Artificial dataset created by:

```
postscript("Class.Multilevel/Images/essex_multilevel_A6.fig01.ps")
beta <- 3; alpha <- 3*(1:5)
x <- runif(n=50,min=0,max=10)
y <- jitter( rep(1,50) %o% alpha + x*beta, factor=3 )
par(bg="slategray",mar=c(0,0,0,0))
plot(c(0,max(x)),c(0,30),type="n",xaxt="n",yaxt="n")
for (i in 1:ncol(y)) segments(0,alpha[i], max(x),
    lm(y[,i] ~ x)$coef %*% c(1,beta),col=colors()[619+i], lwd=3 )
dev.off()
```



Varying-Slope Models

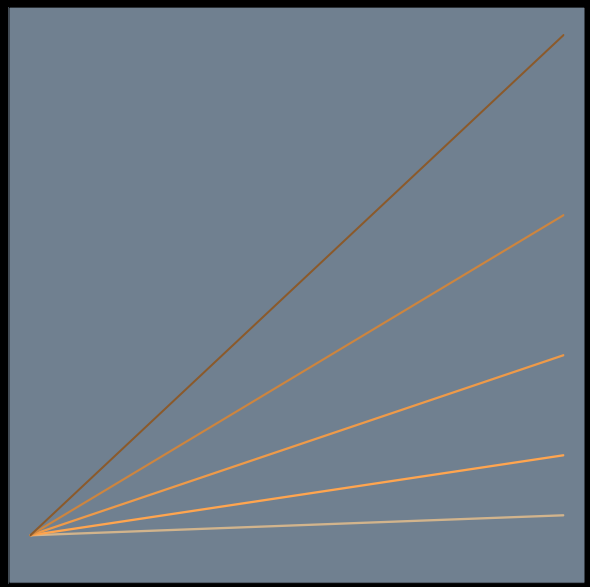
- In the linear context, V-S models include commonality of group but distinct slopes.
- Formally for $i = 1, \ldots, n$:

$$y_i = \beta_0 + \beta_{ij1} X_i + \epsilon_i$$

  where ij indicates that the ith case gets slope j.

- Artificial dataset created by:

```
postscript("Class.Multilevel/Images/essex_multilevel_A6.fig02.ps")
alpha <- 3; beta <- 1.5*(1:5)
x <- runif(n=50,min=0,max=10)
y <- jitter( alpha + x * (rep(1,50) %o% beta) , factor=3 )
par(bg="slategray",mar=c(0,0,0,0))
plot(c(0,max(x)),c(0,60),type="n",xaxt="n",yaxt="n")
for (i in 1:ncol(y)) segments(0,alpha, max(x),
    lm(y[,i] ~ x)$coef %*% c(1,beta[i]),col=colors()[619+i], lwd=3 )
dev.off()
```



Varying-Intercept, Varying-Slope Models

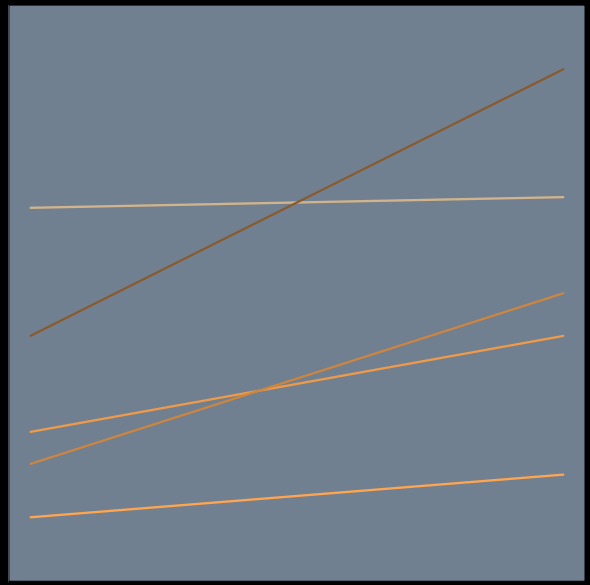
- In the linear context, V-I/V-S models provide interactions between $\mathbf{X}$ and the group designations.
- Formally for $i = 1, \ldots, n$:

$$y_i = \beta_{ij0} + \beta_{ij1} X_i + \epsilon_i$$

  where ij indicates that the ith case gets designator j.

- Artificial dataset created by:

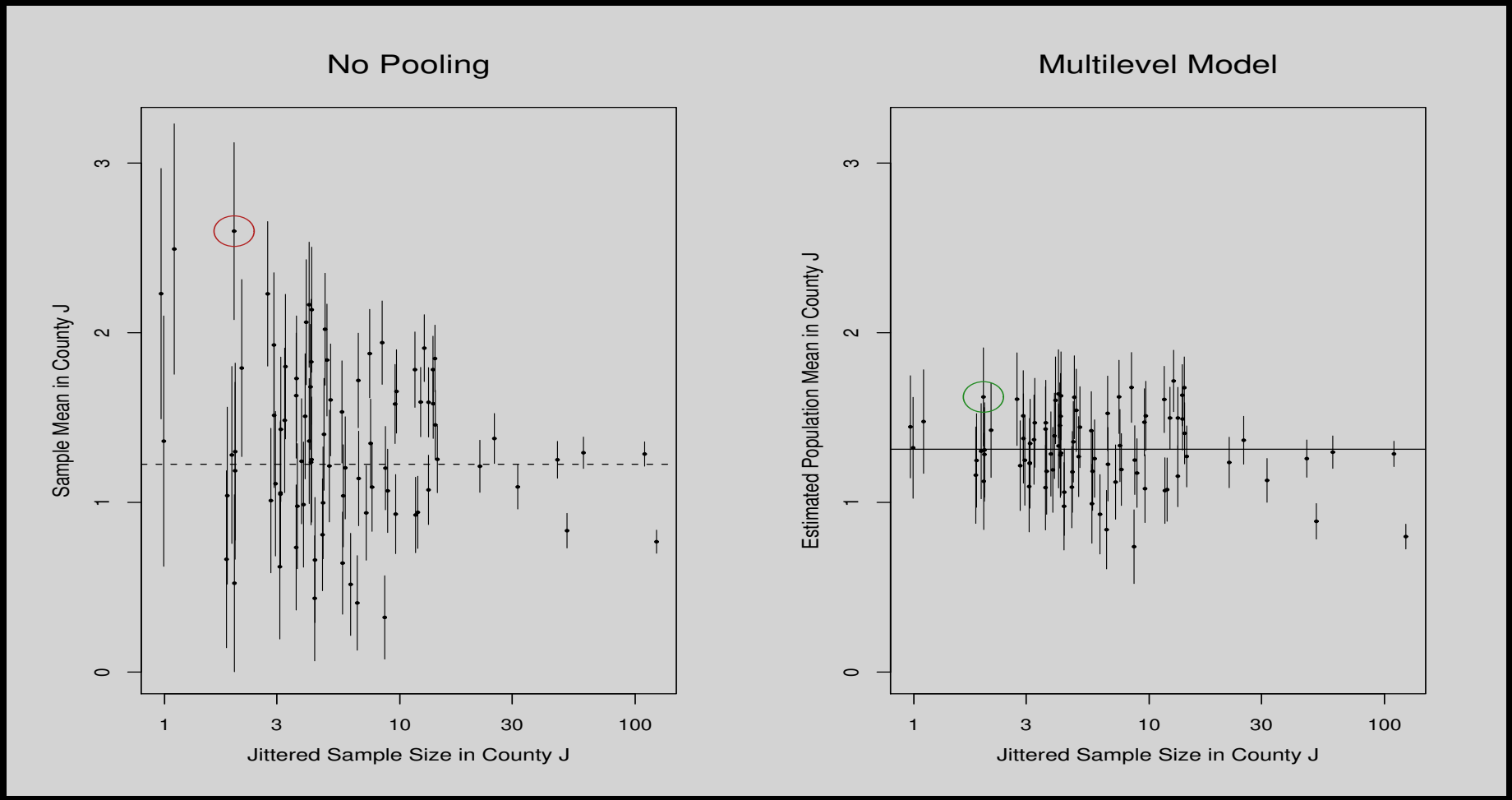
```
postscript("Class.Multilevel/Images/essex_multilevel_A6.fig03.ps")
alpha <- c(33,4,12,9,21); beta <- 1*(1:5)
x <- runif(n=50,min=0,max=10)
y <- jitter( rep(1,50) %o% alpha + x * (rep(1,50) %o% beta) , factor=3 )
par(bg="slategray",mar=c(0,0,0,0))
plot(c(0,max(x)),c(0,50),type="n",xaxt="n",yaxt="n")
for (i in 1:ncol(y)) segments(0,alpha[i], max(x),
    lm(y[,i] ~ x)$coef %*% c(1,beta[i]),col=colors()[619+i], lwd=3 )
dev.off()
```



Back to Pooling

- Complete-Pooling: excluding categorical predictors completely (no hierarchy).
- This ignores (possibly important) variation between categories.
- No-Pooling: estimating separate models for each level of the categorical predictors.
- This overstates variation between categories, making them look more different than they really are (unless the categories are not meaningful).
- Multilevel Models: a compromise between these two extremes that captures within category uniqueness and between category similarities.
- Running example from Gelman & Hill: Radon gas by county ($J = 85$) in Minnesota.

Figure 12.1 from Gelman & Hill



Replicating Gelman & Hill Figure 12.1, Cleaned-Up Gelman Code

```
# SETUP IN R, see: http://jeffgill.org/code/_intro4.R
srrs2 <- read.table("https://jeffgill.org/wp-content/uploads/2024/08/srrs2.txt",
    header=TRUE, sep=",")

( mn <- srrs2$state=="MN" )
[5989]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE
[6001] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[6013] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE

( radon <- srrs2$activity[mn] )
 [1]  2.2  2.2  2.9  1.0  3.1  2.5  1.5  1.0  0.7  1.2  1.2  1.3  1.4  0.4  1.1
[16]  4.5  1.3  2.1  5.9  3.3  1.8  5.4  6.3  1.9  6.6  3.1  6.8  7.0  7.7  5.2
[31]  4.5  4.4  2.8  8.1  1.6  4.2  5.4  4.0  2.3  2.9  1.4  3.3  2.9  1.8  0.2

( log.radon <- log(ifelse (radon==0, .1, radon)) )
 [1]  0.78846  0.78846  1.06471  0.00000  1.13140  0.91629  0.40547  0.00000
 [9] -0.35667  0.18232  0.18232  0.26236  0.33647 -0.91629  0.09531  1.50408
[17]  0.26236  0.74194  1.77495  1.19392  0.58779  1.68640  1.84055  0.64185
```

Replicating Gelman & Hill Figure 12.1, Cleaned-Up Gelman Code

```
( floor <- srrs2$floor[mn] )       # 0 FOR BASEMENT, 1 FOR FIRST FLOOR
 [1] 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[38] 0 0 0 0 0 0 1 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 1 1 1 0 0 1 0 0 0 0 0
[75] 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0 1 0 0 0 0

n <- length(radon)
y <- log.radon
x <- floor

# NOW CREATE MAPPING OF COUNTY NAMES TO 919 HOUSES
( county.name <- as.vector(srrs2$county[mn]) )
[1] "AITKIN              " "AITKIN              " "AITKIN              "
[4] "AITKIN              " "ANOKA              " "ANOKA              "
[7] "ANOKA               " "ANOKA              " "ANOKA              "

( uniq <- unique(county.name) )  # CREATE UNIQUE VERSION
[1] "AITKIN              " "ANOKA              " "BECKER              "
[4] "BELTRAMI            " "BENTON            " "BIG STONE          "
[7] "BLUE EARTH          " "BROWN              " "CARLTON            "
```

Looking at the Identifier for the Group-Level

```
J <- length(uniq)                # J GETS 85
county <- rep(NA, n)             # WILL MAP 919 -> 85
for (i in 1:J)  county[county.name==uniq[i]] <- i

county
  [1]  1  1  1  1  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2
 [22]  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2
 [43]  2  2  2  2  2  2  2  2  2  2  2  2  2  3  3  3  4  4  4  4
 [64]  4  4  4  5  5  5  5  6  6  6  7  7  7  7  7  7  7  7  7  7  7
 [85]  7  7  7  8  8  8  8  9
    :
[841] 79 80 80 80 80 80 80 80 80 80 80 80 80 80 80 80 80 80 80 80
[862] 80 80 80 80 80 80 80 80 80 80 80 80 80 80 80 80 80 80 80 80
[883] 80 80 80 80 80 81 81 81 82 83 83 83 83 83 83 83 83 83 83 83
[904] 83 84 84 84 84 84 84 84 84 84 84 84 84 84 85 85
```

Radon Data Structures, *How I Would Do It*

```
srrs2 <- read.table("Class.Multilevel/examples/radon/srrs2.dat", header=TRUE, sep=",")
mn <- srrs2$state=="MN"
mn.df <- data.frame( "log.radon"=log(ifelse(radon==0,0.1,srrs2$activity[mn])),
                     "floor"=srrs2$floor[mn],
                     "county.name"=as.vector(srrs2$county[mn]))
                     "county"=rep(NA,sum(mn) )
n <- nrow(mn.df)
uniq <- unique(mn.df$county.name)
county <- rep(NA, n)
for (i in 1:length(uniq)) county[county.name==uniq[i]] <- i
mn.df$county <- county
mn.df[916:n,]
    log.radon floor          county.name county
916    1.5041     0 WRIGHT                    84
917    1.6094     0 WRIGHT                    84
918    1.3083     0 YELLOW MEDICINE           85
919    1.0647     0 YELLOW MEDICINE           85
```

Replicating Gelman & Hill Figure 12.1

```
( sample.size <- as.vector(table (county)) )
 [1]   4  52   3   7   4   3  14   4  10   6   5   4   6  14   4   2   4  12  63
[20]   3   9   6   2   9  14 105   6   5   3  11   5   4   4   3   7   2   9   4
[39]   5   4   8   1   9   7  13   5   2   9  13   1   4   3   3  23   8   3   6
[58]   4   4   2  32   5   3  11   2  14  13   8   4 116  25  10   2   4   3   4
[77]   7   5   4  46   3   1  13  13   2

sample.size.jittered <- sample.size*exp(runif (J, -.1, .1))
ybarbar = mean(y)
```

Replicating Gelman & Hill Figure 12.1, County Level Means

```
( cty.mns = tapply(y,county,mean) )
      1       2       3       4       5       6       7       8       9      10
0.66041 0.83325 1.04834 1.14099 1.25244 1.51301 1.90923 1.62931 0.93098 1.20364
     11      12      13      14      15      16      17      18      19      20
1.40113 1.73025 1.03872 1.78252 0.97770 0.66486 0.73439 0.94141 1.29256 1.80032
21   22      23      24      25      26      27      28      29      30
1.65393 0.51670 1.03972 1.94150 1.84779 1.28510 1.53318 0.80927 1.05600 0.92576
     31      32      33      34      35      36      37      38      39      40
2.02057 1.23629 2.06187 1.11008 0.40746 2.59870 0.32203 1.50745 1.60396 2.13567
     41      42      43      44      45      46      47      48      49      50
1.87708 1.36098 1.20194 0.93805 1.07371 1.21461 0.52366 1.06806 1.59126 2.49321
     51      52      53      54      55      56      57      58      59      60
2.16504 1.92769 1.01062 1.21277 1.34779 0.62074 0.64209 1.68079 1.36151 1.27939
     61      62      63      64      65      66      67      68      69      70
1.09118 1.83890 1.43052 1.78233 1.29912 1.25400 1.58142 1.09002 1.24245 0.76825
     71      72      73      74      75      76      77      78      79      80
1.37656 1.57990 1.79176 0.98704 1.48354 1.82830 1.71875 0.99747 0.43475 1.25120
     81      82      83      84      85
2.22917 2.23001 1.45664 1.58993 1.18652
```

Replicating Gelman & Hill Figure 12.1, County Level Variances

```
( cty.vars = tapply(y,county,var) )
     1      2      3      4      5      6      7      8      9     10     11
0.2108 0.5933 0.5630 0.9363 0.1801 0.2663 0.3058 0.3696 0.3782 3.6210 0.2542
    12     13     14     15     16     17     18     19     20     21     22
0.8758 0.7139 1.0206 0.5478 0.2157 4.4840 0.4471 0.5660 0.0234 0.1233 2.1476
    23     24     25     26     27     28     29     30     31     32     33
0.0305 0.6508 0.5050 0.5047 0.3049 0.5101 0.0620 0.3414 0.4748 0.3997 0.2605
    34     35     36     37     38     39     40     41     42     43     44
1.1536 0.1403 0.0605 0.6830 0.3150 0.2788 0.4543 0.1040     NA 2.2389 1.5932
    45     46     47     48     49     50     51     52     53     54     55
1.9547 0.2425 2.9612 0.1513 0.8843     NA 0.3102 0.0937 0.8768 0.5784 0.6861
    56     57     58     59     60     61     62     63     64     65     66
2.5477 0.2440 0.7819 0.7948 0.0062 0.4463 0.8132 0.3559 0.3196 1.3749 0.4642
    67     68     69     70     71     72     73     74     75     76     77
0.9455 0.1416 0.3208 0.6398 0.4926 0.3582 0.1655 0.2338 0.4164 0.6883 0.6998
    78     79     80     81     82     83     84     85
0.6236 1.7831 0.6345 0.2020     NA 1.0859 0.3590 0.0297
```

Replicating Gelman & Hill Figure 12.1, County Level Standard Deviations

- Recall that we are assuming (for now) that σ_y^2 is constant with groups and across groups.
- Therefore we will mean the variances per county as just calculated, then scale them by county size:

$$SD_{\alpha_j} = \frac{\sigma_y}{n_j}.$$

- So this is the standard error of the mean in county j keeping the constant variance assumption but accounting for differing sample size.
- For the `NA` values in the variance list, we assign the Minnesota mean, using:

```
mean( sqrt(cty.vars[!is.na(cty.vars)]) )
[1] 0.73822
```

Replicating Gelman & Hill Figure 12.1, County Level Standard Deviations

```
( cty.sds = mean( sqrt(cty.vars[!is.na(cty.vars)]) )/sqrt(sample.size) )
 [1] 0.369112 0.102373 0.426214 0.279023 0.369112 0.426214 0.197299 0.369112
 [9] 0.233447 0.301379 0.330144 0.369112 0.301379 0.197299 0.369112 0.522003
[17] 0.369112 0.213107 0.093008 0.426214 0.246075 0.301379 0.522003 0.246075
[25] 0.197299 0.072043 0.301379 0.330144 0.426214 0.222583 0.330144 0.369112
[33] 0.369112 0.426214 0.279023 0.522003 0.246075 0.369112 0.330144 0.369112
[41] 0.261002 0.738224 0.246075 0.279023 0.204747 0.330144 0.522003 0.246075
[49] 0.204747 0.738224 0.369112 0.426214 0.426214 0.153930 0.261002 0.426214
[57] 0.301379 0.369112 0.369112 0.522003 0.130501 0.330144 0.426214 0.222583
[65] 0.522003 0.197299 0.204747 0.261002 0.369112 0.068542 0.147645 0.233447
[73] 0.522003 0.369112 0.426214 0.369112 0.279023 0.330144 0.369112 0.108845
[81] 0.426214 0.738224 0.204747 0.204747 0.522003
```

Replicating Gelman & Hill Figure 12.1

```
postscript("Class.Multilevel/figure12.1.ps")
par(mfrow=c(1,2),mar=c(5,5,5,3),bg="lightgray")

# LEFT PANEL
plot(sample.size.jittered, cty.mns, cex.lab=.9, cex.axis=1, xlab="", ylab="",
      pch=20, log="x", cex=.5, ylim=c(0,3.2), yaxt="n", xaxt="n")
mtext(side=1,line=2.5,"Jittered Sample Size in County J")
mtext(side=2,line=2.5,"Sample Mean in County J")
mtext(side=3,line=1.5,"No-Pooling",cex=1.5)
axis(1, c(1,3,10,30,100), cex.axis=.9); axis(2, seq(0,3), cex.axis=.9)
# LOOP THAT PRODUCES +/- 1 ERROR BARS
for (j in 1:J) lines(rep(sample.size.jittered[j],2),
                     cty.mns[j] + c(-1,1)*cty.sds[j], lwd=.5)
abline(h=ybarbar,lty=2)
# HIGHLIGHT CASE NUMBER 36
points(sample.size.jittered[36],cty.mns[36],cex=4,col="firebrick")
```

Replicating Gelman & Hill Figure 12.1

```
# RIGHT PANEL
plot(sample.size.jittered, mlm.radon.mean, cex.lab=.9, cex.axis=1,
      xlab="", ylab="", pch=20, log="x", cex=.5,
      ylim=c(0,3.2), yaxt="n", xaxt="n")
mtext(side=1,line=2.5,"Jittered Sample Size in County J")
mtext(side=2,line=2.5,"Estimated Population Mean in County J")
mtext(side=3,line=1.5,"Multilevel Model",cex=1.5)
axis(1, c(1,3,10,30,100), cex.axis=.9); axis(2, seq(0,3), cex.axis=.9)
# LOOP THAT PRODUCES +/- 1 ERROR BARS
for (j in 1:J) lines(rep(sample.size.jittered[j],2),
         mlm.radon.mean[j] + c(-1,1)*mlm.radon.sd[j], lwd=.5)
abline(h=mean(mlm.radon.mean))
# HIGHLIGHT CASE NUMBER 36
points(sample.size.jittered[36],mlm.radon.mean[36],cex=4,col="forestgreen")
dev.off()
```

Partial Pooling Estimates with No Explanatory Variables

- Panel 2 of Figure 12.1 gives the average log radon level α_j by county $j = 1, \ldots, J$, plotted by n_j which also varies by county.

- These were produced from the multilevel model estimated with `JAGS` , but are approximated by:

$$\alpha_j \approx \frac{\frac{n_j}{\sigma_y^2}\bar{y}_j + \frac{1}{\sigma_\alpha^2}\bar{y}_{\text{all}}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}$$

where:

$\bar{y}_j$	unpooled estimate for county j
$\bar{y}_{\text{all}}$	completely pooled estimate
σ_y^2	within-county variance (assumed equal for now)
σ_α^2	variance among the mean estimates

Partial Pooling Estimates with No Explanatory Variables

- County Sample Size Consequences from:

$$\alpha_j \approx \frac{\frac{n_j}{\sigma_y^2}\bar{y}_j + \frac{1}{\sigma_\alpha^2}\bar{y}_{\text{all}}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}$$

Partial Pooling Estimates with No Explanatory Variables

- County Sample Size Consequences from:

$$\alpha_j \approx \frac{\frac{n_j}{\sigma_y^2}\bar{y}_j + \frac{1}{\sigma_\alpha^2}\bar{y}_{\text{all}}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}$$

- Averages from counties with smaller sample sizes contribute less and are closer to the state average, and county estimate equal to the state average for $n_j = 0$.

Partial Pooling Estimates with No Explanatory Variables

- County Sample Size Consequences from:

$$\alpha_j \approx \frac{\frac{n_j}{\sigma_y^2}\bar{y}_j + \frac{1}{\sigma_\alpha^2}\bar{y}_{\text{all}}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}$$

- Averages from counties with smaller sample sizes contribute less and are closer to the state average, and county estimate equal to the state average for $n_j = 0$.
- Averages from counties with larger sample sizes contribute more and pull the state average towards them, and state average equal to the county estimate for $n_j = \infty$.

Partial Pooling Estimates with No Explanatory Variables

- County Sample Size Consequences from:

$$\alpha_j \approx \frac{\frac{n_j}{\sigma_y^2}\bar{y}_j + \frac{1}{\sigma_\alpha^2}\bar{y}_{\text{all}}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}$$

- Averages from counties with smaller sample sizes contribute less and are closer to the state average, and county estimate equal to the state average for $n_j = 0$.
- Averages from counties with larger sample sizes contribute more and pull the state average towards them, and state average equal to the county estimate for $n_j = \infty$.
- Output from `JAGS` that generated the second panel of Figure 12.1:

	Mean	SD	NaiveSE	TimeseriesSE
sigma.y	0.7991	0.01956	0.0001956	0.0002304
a1	1.0602	0.25501	0.0025501	0.0031325
mu.a	1.3136	0.05011	0.0005011	0.0009859
sigma.a	0.3171	0.04888	0.0004888	0.0014624

where `NaiveSE` $= \sqrt{\text{sample variance}}/\sqrt{n}$ and: `TIMEseriesSE` $= \sqrt{\text{spectral density var}}/\sqrt{n} =$ asymptotic SE.

Partial Pooling Estimates with Explanatory Variables

- We can also re-express this as a weighted average of the no-pooling estimate and the pooled estimate mean:

$$\alpha_j \approx \frac{\frac{n_j}{\sigma_y^2}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}(\bar{y}_j - \beta \bar{X}_j) + \frac{\frac{1}{\sigma_\alpha^2}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}\mu_\alpha$$

  where the first term is a weighted no-pooled regression of the jth case and the second term is a weighted completely pooled mean.

Partial Pooling Estimates with Explanatory Variables

- We can also re-express this as a weighted average of the no-pooling estimate and the pooled estimate mean:

$$\alpha_j \approx \frac{\frac{n_j}{\sigma_y^2}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}(\bar{y}_j - \beta \bar{X}_j) + \frac{\frac{1}{\sigma_\alpha^2}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}\mu_\alpha$$

  where the first term is a weighted no-pooled regression of the jth case and the second term is a weighted completely pooled mean.

- This shows the *shrinkage* that occurs when some n_j gets large relative to the others (sometimes called the "Manhattan Effect").

Partial Pooling Estimates with Explanatory Variables

- We can also re-express this as a weighted average of the no-pooling estimate and the pooled estimate mean:

$$\alpha_j \approx \frac{\frac{n_j}{\sigma_y^2}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}(\bar{y}_j - \beta \bar{X}_j) + \frac{\frac{1}{\sigma_\alpha^2}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}\mu_\alpha$$

  where the first term is a weighted no-pooled regression of the jth case and the second term is a weighted completely pooled mean.

- This shows the *shrinkage* that occurs when some n_j gets large relative to the others (sometimes called the "Manhattan Effect").

- When $\sigma_\alpha^2 \to 0$, then the results move toward the complete-pooling model.

Partial Pooling Estimates with Explanatory Variables

- We can also re-express this as a weighted average of the no-pooling estimate and the pooled estimate mean:

$$\alpha_j \approx \frac{\frac{n_j}{\sigma_y^2}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}(\bar{y}_j - \beta \bar{X}_j) + \frac{\frac{1}{\sigma_\alpha^2}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}\mu_\alpha$$

  where the first term is a weighted no-pooled regression of the jth case and the second term is a weighted completely pooled mean.

- This shows the *shrinkage* that occurs when some n_j gets large relative to the others (sometimes called the "Manhattan Effect").

- When $\sigma_\alpha^2 \to 0$, then the results move toward the complete-pooling model.

- When $\sigma_\alpha^2 \to \infty$, then the results move toward the no-pooling model.

Contrasting Pooling Approaches with Floor(0,1) Explanatory Variable

- Complete-pooling: $y_i = \alpha_i + \beta X_i + \epsilon_i$, versus (approximated) No-pooling: $y_{ij} = \alpha_{ij} + \beta_{ij} X_i + \epsilon_{ij}$.

```
lm.pooled <- lm(y ~ x)
summary(lm.pooled)
```

```
            Estimate      SE t-stat
(Intercept)  1.32674 0.02972 44.640
          x -0.61339 0.07284 -8.421
```

```
RSE: 0.8226 on 917 df
R-Squared: 0.0717, Adj.R-Sqr: 0.0707
F: 70.91, 1 and 917 df,  p < 2.2e-16
```

```
lm.unpooled.approx <-
                lm(y ~ x + factor(county) - 1)
summary(lm.unpooled.approx)
```

```
                      Estimate        SE  t-stat
x                     -0.72054   0.07352  -9.800
                                      :
factor(county)81      2.709533   0.439461  6.1656 1.094e-09
n=1 -->               2.230014   0.756420  2.9481 0.0032865
factor(county)83      1.622923   0.210478  7.7107 3.567e-14
factor(county)84      1.645354   0.209869  7.8399 1.377e-14
n=2 -->               1.186522   0.534869  2.2183 0.0268006
```

```
RSE: 0.7564 on 833 df
R-Squared: 0.7671, Adj.R-Sqr: 0.7431
F: 31.91, 86 and 833 df, p < 2.2e-16
```

Multilevel Model Fitting: Varying-Intercept, Individual-level Explanatory Variable

- Includes a state-wide constant, a state-wide covariate, and variability by county, $y_i = \alpha + \alpha_{ij} + \beta X_i + \epsilon_i$, fit by:

```
M1 <- glmer(y ~ x + (1 | county))
summary(M1)

Random effects:
 Groups   Name        Variance Std.Dev.
 county   (Intercept) 0.108    0.328
 Residual             0.571    0.756
number of obs: 919, groups: county, 85

Fixed effects:
            Estimate Std. Error t value
(Intercept)   1.4616     0.0516   28.34
x            -0.6930     0.0704   -9.84
```

  where we could have random slopes as well with `(1+x | county)`.

- Meaning that: $\sigma^2_\alpha = 0.108$, $\sigma^2_y = 0.571$.

Modeling Notes

- Faraway (p.196) notes that the simplest model that has a random effect is a one-way ANOVA given by:

$$y_{ij} = \mu + \alpha_i + \epsilon_{ij}$$

, for $i = 1, \ldots a$ groups and $j = 1, \ldots n$ individuals (backward notation from convention).

- Even this simple setup produces two variances: σ^2_α the variance between groups, and σ^2_ϵ the variance of the residuals (individual cases).

- Since all variance not soaked up by the group definitions in the model and thus falling to indvidual variance the "game" is comparing these two values, which is sometimes done with the *intraclass correlation coefficient*:

$$\text{ICC} = \frac{\sigma^2_\alpha}{\sigma^2_\alpha + \sigma^2_\epsilon}$$

- Limitations of the ICC: no distributional property, needs to be calculated multiple times for models with many hierarchies, and the "rules of thumb" in published work are wrong.

- These specifications can also be non-nested (crossed effects in Faraway) or nested (multilevel in Faraway), or both.

Estimation Notes

- However, ML estimation can be problematic, so we need to *Restricted Maximum Likelihood* (REML) estimation (Bartlett 1937).
- REML uses a likelihood function calculated from a transformed set of data so that some parameters have no effect on the estimation on the others.
- Steps here:
  1. find a linear transformation k such that $k'\mathbf{X} = 0$, so $k'\mathbf{y} \sim N(0, k'\mathbf{V}k)$.
  2. run MLE on this model to get $\hat{\mathbf{D}}$, which no longer has any fixed effects
  3. then estimate the fixed effects with ML in the normal way.
- So some parameters are estimated serially in isolation.
- REML deviance is always a little higher due to the two stages of the estimation process.
- The default with `glmer` is REML but it can be turned off with `REML=FALSE` in the modeling statement.

REML

The Annals of Statistics
1996, Vol. 24, No. 1, 255–286

REML ESTIMATION: ASYMPTOTIC BEHAVIOR AND RELATED TOPICS

BY JIMING JIANG

University of California, Berkeley

The restricted maximum likelihood (REML) estimates of dispersion parameters (variance components) in a general (non-normal) mixed model are defined as solutions of the REML equations. In this paper, we show the REML estimates are consistent if the model is asymptotically identifiable and infinitely informative under the (location) invariant class, and are asymptotically normal (A.N.) if in addition the model is asymptotically nondegenerate. The result does not require normality or boundedness of the rank p of design matrix of fixed effects. Moreover, we give a necessary and sufficient condition for asymptotic normality of Gaussian maximum likelihood estimates (MLE) in non-normal cases. As an application, we show for all unconfounded balanced mixed models of the analysis of variance the REML (ANOVA) estimates are consistent; and are also A.N. provided the models are nondegenerate; the MLE are consistent (A.N.) if and only if certain constraints on p are satisfied.

Multilevel Model Fitting: Varying-Intercept, Individual-level Explanatory Variable

- The county-averaged model comes from the fixed effects:

$$y = 1.4616 - 0.6930X$$

  which can be isolated with the command `fixef(M1)`.

- The individual-level models are produced with (notice the heavy rounding):

```
coef(M1)
   (Intercept)     X
1         1.19 -0.69
2         0.93 -0.69
3         1.48 -0.69
:
83        1.57 -0.69
84        1.59 -0.69
85        1.39 -0.69
```

  so the first county model is:

$$y_1 = 1.19 - 0.69X$$

Multilevel Model Fitting: Varying-Intercept, Individual-level Explanatory Variable

- We can also see how much the intercept is shifted for each case from $\alpha = 1.4616$:

```
ranef(M1)
   (Intercept)
1       -0.270
2       -0.534
3        0.018
:
83       0.110
84       0.129
85      -0.075
```

- There are also functions for directly obtaining standard errors:

```
se.fixef(M1)
(Intercept)           x
   0.051573    0.070431
```

```
se.ranef(M1)
$county
          [,1]
[1,] 0.247785
[2,] 0.099827
[3,] 0.262286
:
```

Multilevel Model Fitting: Varying-Intercept, Individual-level Explanatory Variable

- The latter can be used to produce a 95% confidence interval for X:

```
fixef(M1)["x"] + c(-1.96,1.96)*se.fixef(M1)["x"]
-0.83 -0.55
```

- And a 95% confidence interval for the intercept in county 55:

```
coef(M1)$county[55,1] + c(-1.96,1.96)*se.ranef(M1)$county[55]
1.1 2.0
```

- Also a 95% confidence interval for the deviation from the average in the intercept in county 55:

```
ranef(M1)$county[55,] + c(-1.96,1.96)*se.ranef(M1)$county[55]
-0.32  0.50
```

Adding Group-Level Explanatory Variables, Some Data Setup

- There is one uncomfortable aspect of estimating this model: `glmer` is limited in how it specifies group-level variables.
- So sometimes we have to trick the function with a pre-binned version of the variable of interest: `u.full`.
- The additional G&H setup code follows (see their explanation at the bottom of page 266).
- Get county-level information from my website that links FIPS to Uranium:

```
cty <- read.table("http://jeffgill.org/data/cty.dat", header=TRUE, sep=",")
names(cty)
[1] "stfips" "ctfips" "st"    "cty"   "lon"   "lat"   "Uppm"
```

  (FIPS = Federal Information Processing Standards Codes that link geographic units, counties and county equivalents).

- Multiply the state FIPS values by 1000 and add the county-level FIPS values:

```
srrs2.fips <- srrs2$stfips*1000 + srrs2$cntyfips
```

Adding Group-Level Explanatory Variables, Some Data Setup

- Create a variable for scaled FIPS:

```
usa.fips <- 1000*cty[,"stfips"] + cty[,"ctfips"]
```

- Now get the county unique FIPS for Minnesota only using `match`:

```
( usa.rows <- match (unique(srrs2.fips[mn]), usa.fips) )
 [1] 1327 1328 1329 1330 1331 1332 1333 1334 1335 1336 1338 1339 1340 1341 1342
[16] 1343 1344 1345 1346 1347 1348 1349 1350 1351 1352 1354 1355 1356 1357 1358
[31] 1359 1360 1361 1362 1363 1364 1366 1367 1368 1369 1370 1372 1373 1374 1371
[46] 1375 1376 1377 1378 1379 1380 1381 1382 1383 1384 1385 1386 1387 1388 1389
[61] 1390 1392 1393 1394 1395 1396 1398 1399 1400 1397 1401 1402 1403 1404 1405
[76] 1406 1407 1408 1409 1410 1411 1412 1413 1414 1415
```

- Federal Information Processing Standards are public standards developed by the US federal government for non-military computer use by government agencies and contractors, including county and state identifying FIPS codes.

Adding Group-Level Explanatory Variables, Some Data Setup

- Use these Minnesota FIPS to get the Uranium parts per million measurement:

```
( uranium <- cty[usa.rows,"Uppm"] )
 [1] 0.50205 0.42856 0.89274 0.55247 0.86685 1.47264 1.31208 1.31993 0.71726
[10] 1.10061 0.54432 1.31480 0.47935 1.41027 0.94190 0.60351 1.40433 0.53079
[19] 0.97614 1.30194 1.16849 1.34316 1.51424 1.25133 1.21727 0.90799 1.65455
[28] 0.66992 0.47148 0.51512 1.36209 0.94801 1.11598 0.99223 0.41402 1.36483
[37] 0.50078 0.50575 1.21464 1.56034 1.48399 1.16137 1.01386 1.18041 1.15076
[46] 1.02424 0.81054 0.91099 1.29814 1.49011 1.28152 1.49998 1.30372 1.27526
[55] 0.81487 0.92865 0.84934 1.61382 1.30488 1.32465 0.65833 1.44242 1.46313
[64] 1.21306 1.69558 0.80893 1.06515 0.50488 1.26771 0.62209 1.12344 1.30971
[73] 1.60124 1.37167 0.95424 1.64476 1.16193 0.51067 1.23666 0.86288 1.20110
[82] 1.26622 1.58917 0.91391 1.42659
```

Adding Group-Level Explanatory Variables, Some Data Setup

- Log the uranium measurement by county:

```
( u <- log(uranium) )
 [1] -0.6890 -0.8473 -0.1135 -0.5934 -0.1429  0.3871  0.2716  0.2776 -0.3323
[10]  0.0959 -0.6082  0.2737 -0.7353  0.3438 -0.0599 -0.5050  0.3396 -0.6334
[19] -0.0241  0.2639  0.1557  0.2950  0.4149  0.2242  0.1966 -0.0965  0.5035
[28] -0.4006 -0.7519 -0.6633  0.3090 -0.0534  0.1097 -0.0078 -0.8818  0.3110
[37] -0.6916 -0.6817  0.1944  0.4449  0.3947  0.1496  0.0138  0.1659  0.1404
[46]  0.0240 -0.2101 -0.0932  0.2609  0.3988  0.2480  0.4055  0.2652  0.2432
[55] -0.2047 -0.0740 -0.1633  0.4786  0.2661  0.2811 -0.4181  0.3663  0.3806
[64]  0.1931  0.5280 -0.2120  0.0631 -0.6834  0.2372 -0.4747  0.1164  0.2698
[73]  0.4708  0.3160 -0.0468  0.4976  0.1501 -0.6720  0.2124 -0.1475  0.1832
[82]  0.2360  0.4632 -0.0900  0.3553
```

Adding Group-Level Explanatory Variables, Some Data Setup

- Now create variable that has replicate county measurements so we can put it at the base level for `glmer`, but it is really a county-level variable:

```
( u.full <- u[county] )
  [1] -0.6890476 -0.6890476 -0.6890476 -0.6890476 -0.8473129
  [6] -0.8473129 -0.8473129 -0.8473129 -0.8473129
    :
[911] -0.0900243 -0.0900243 -0.0900243 -0.0900243 -0.0900243
[916] -0.0900243 -0.0900243  0.3552870  0.3552870
```

Adding Group-Level Explanatory Variables

- We will use uranium (`u.full <- u[county]`), 919 long but 85 unique values:

```
M2 <- glmer(y ~ x + u.full + (1 | county))

# LOOK AT THE FIXED EFFECTS ONLY
fixef(M2)
(Intercept)           x      u.full
    1.46576    -0.66824     0.72027

# COMBINE THE AGGREGATE COUNTY LEVEL EFFECTS FOR URANIUM, RANDOM INTERCEPTS
( a.hat.M2 <- fixef(M2)[1] + fixef(M2)[3]*u + as.vector(ranef(M2)$county) )
   (Intercept)
1      0.94882
2      0.86671
3      1.39646
:
83     1.73228
84     1.48650
85     1.67972
```

Adding Group-Level Explanatory Variables

- Where `as.vector(ranef(M2)$county)` returns `-0.02064, 0.01125, 0.01242,...`.
- Label the floor effect:

```
( b.hat.M2 <- fixef(M2)[2] )
       x
-0.66824
```

- Important Note: your results will differ slightly due (1) heavy-handed rounded by G&H, (2) changes in the `glmer` command over time.

Adding Group-Level Explanatory Variables

- Results from the unmodeled coefficients modeled coefficients add up to the coefficient estimates.
- Look at a house in county 85 from `coef(M2)`, $u_{85} = 0.36$, $X = 0, 1$:

$$\begin{aligned} y_{85} &= \alpha_{85} + \beta X_{85} + \gamma_1 u_{85} \\ &= 1.42 - 0.67 X_{85} + 0.72(0.36) \\ &= \begin{cases} 1.42 - 0.67(0) + 0.72(0.36) = 1.68 & \text{measured in basement} \\ 1.42 - 0.67(1) + 0.72(0.36) = 1.01 & \text{measured on first floor} \end{cases} \end{aligned}$$

- Now start with the fixed effects model and add the group level error:

$$\begin{aligned} y_{85} &= \eta + \eta_{85} + \beta X_{85} + \gamma_1 u_{85} \\ &= 1.47 - 0.04 - 0.67 X_{85} + 0.72(0.36) \\ &= 1.69 - 067 X_{85} \end{aligned}$$

Looking At the Full Summary

```
summary(M2)
Linear mixed model fit by REML
Formula: y ~ x + u.full + (1 | county)
   AIC   BIC logLik deviance REMLdev
  2144  2168  -1067     2123    2134
Random effects:
 Groups   Name        Variance Std.Dev.
 county   (Intercept) 0.0245   0.156
 Residual             0.5752   0.758
Number of obs: 919, groups: county, 85

Fixed effects:
            Estimate Std. Error t value
(Intercept)   1.4658     0.0379    38.6
x            -0.6682     0.0688    -9.7
u.full        0.7203     0.0918     7.8

Correlation of Fixed Effects:
(Intr) x
x      -0.357
u.full  0.145 -0.009
```

Exploiting Updates in glmer

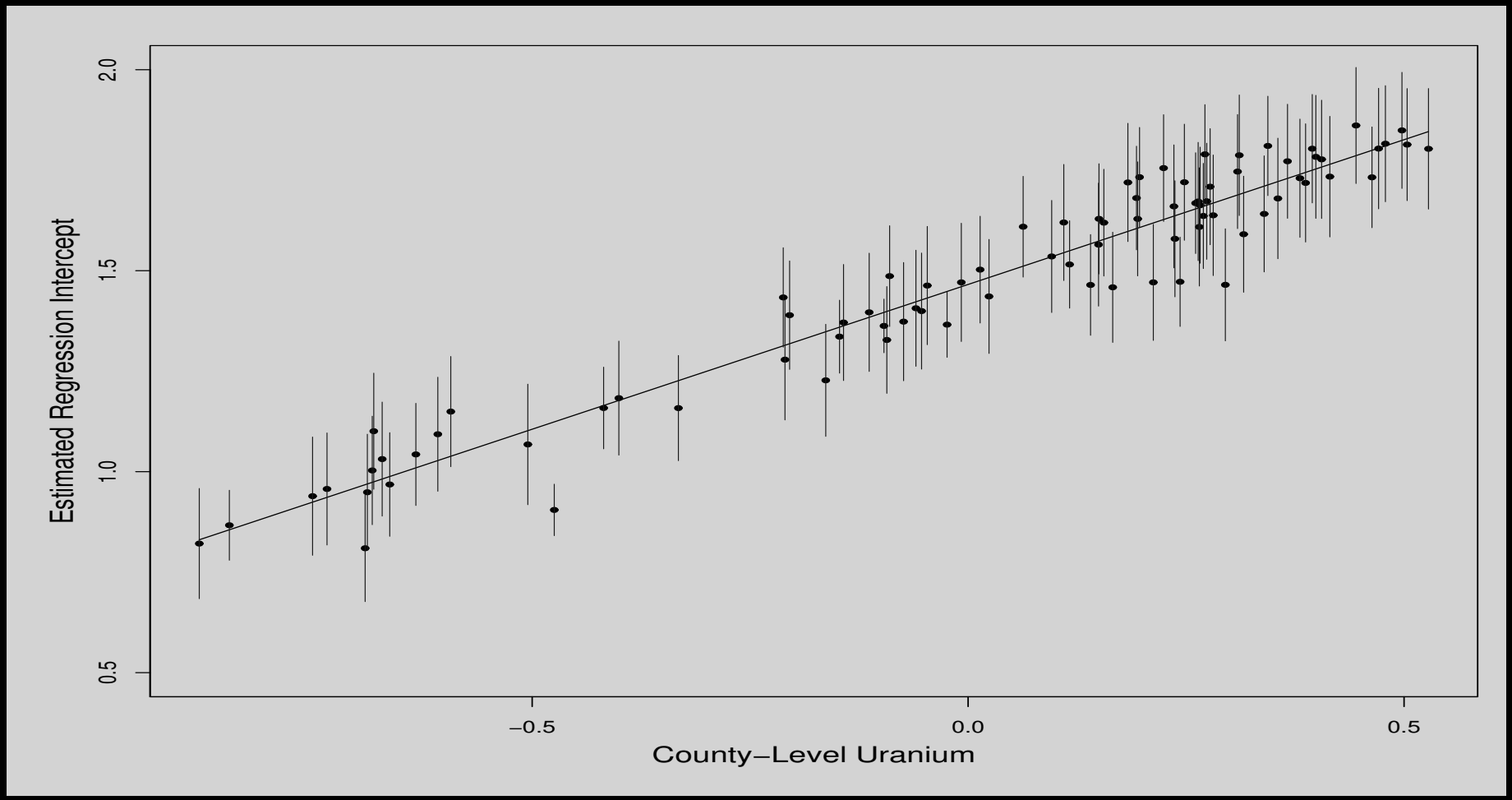
```
M2 <- glmer(y ~ x + (1 + u.full | county))

summary(M2)
Scaled residuals:
   Min     1Q Median     3Q    Max
   -4.353 -0.620  0.006  0.626  3.415

Random effects:
 Groups   Name        Variance Std.Dev. Corr
 county   (Intercept) 0.0107   0.104
          u.full      0.7169   0.847    0.87
          Residual             0.5722   0.756
          Number of obs: 919, groups:  county, 85

Fixed effects:
            Estimate Std. Error t value
(Intercept)   1.4364     0.0398    36.1
x            -0.6772     0.0694    -9.8
```

Gelman & Hill Figure 12.6



R Code for Figure 12.6

```
postscript("Class.Multilevel/figure12.6.ps")
par(mfrow=c(1,1),mar=c(5,5,2,1),bg="lightgray")
plot(u, t(a.hat.M2), cex.lab=1.1, cex.axis=1.1, pch=20, ylab="",xlab="",
    yaxt="n", xaxt="n", ylim=c(0.5,2))
axis(1, seq(-1,1,.5), cex.axis=1.0)
axis(2, seq(0,2.0,.5), cex.axis=1.0)
mtext(side=1,cex=1.3,"County-Level Uranium",line=2.5)
mtext(side=2,cex=1.3,"Estimated Regression Intercept",line=2.5)
curve(fixef(M2)["(Intercept)"] + fixef(M2)["u.full"]*x, lwd=1,
    col="black", add=TRUE)
for (j in 1:J)
    lines(rep(u[j],2), a.hat.M2[j,] + c(-1,1)*se.coef(M2)$county[j,],
        lwd=.5, col="gray10")
dev.off()
```

Varying Intercepts and Varying Slopes

- First level:

$$y_i \sim N(\alpha_{ij} + \beta_{ij} X_i, \sigma_y^2), \qquad i = 1, \ldots, n$$

- Second level:

$$\begin{pmatrix} \alpha_j \\ \beta_j \end{pmatrix} \sim N\left(\begin{pmatrix} \mu_\alpha \\ \mu_\beta \end{pmatrix}, \begin{pmatrix} \sigma_\alpha^2 & \rho\sigma_\alpha\sigma_\beta \\ \rho\sigma_\alpha\sigma_\beta & \sigma_\beta^2 \end{pmatrix} \right) \qquad j = 1, \ldots, J$$

- Where we introduce the new term ρ as the *between group correlation parameter.*
- `x` is not a group-level covariate, but is specified at both levels so that β is allowed to vary at the second level.

```
y ~ 1 + x + (1 + x | county)
```

Implementation in R

```
M3 <- glmer(y ~ 1 + x + (1 + x | county))
summary(M3)
Random effects:
 Groups   Name        Variance Std.Dev. Corr
 county   (Intercept) 0.122    0.349
          x           0.118    0.344    -0.337
 Residual             0.557    0.746
number of obs: 919, groups: county, 85

Fixed effects:
            Estimate Std. Error t value
(Intercept)   1.4628     0.0539   27.15
x            -0.6811     0.0876   -7.78

Correlation of Fixed Effects:       (Intr)   x -0.381
```

Looking At Cases

- And to see the estimated random effects (group-level errors):

```
ranef(M3)
An object of class \ranef.lmer"
[[1]]
   (Intercept)          x
1   -0.3182642  0.1405481
2   -0.5293905 -0.0898034
3    0.0089163  0.0122211
:
83   0.2315155 -0.4702222
84   0.1363534 -0.0516471
85  -0.0839834  0.0279326

is.numeric(as.matrix(ranef(M3)[[1]]))
[1] TRUE
```

- The `as.matrix()` command is required if you use this as input to a numeric function.

Looking At Cases

- Reminder: this is not (yet) a group-level *covariate*: X is specified at both levels so that β is allowed to vary at the second level.
- Information for county 85:

```
fixef(M3)
(Intercept)           x
    1.46277    -0.68109

ranef(M3)
   (Intercept)          x
85  -0.0839834  0.0279326
```

- The estimated regression for county 85 is:

$$\begin{aligned}\hat{y}_j &= (\mu_\alpha + \eta_j^\alpha) + (\mu_\beta + \eta_j^\beta)X, \qquad j = 85 \\ &= (1.46 - 0.08) + (-0.68 + 0.03)X \\ &= 1.38 - 0.65X\end{aligned}$$

Group-Level Covariates

- First level:

$$y_i \sim N(\alpha_{ij} + \beta_{ij} X_i, \sigma_y^2), \qquad i = 1, \ldots, n$$

- Second level:

$$\begin{pmatrix} \alpha_j \\ \beta_j \end{pmatrix} \sim N\left(\begin{pmatrix} \gamma_0^\alpha + \gamma_1^\alpha u_j \\ \gamma_0^\beta + \gamma_1^\beta u_j \end{pmatrix}, \begin{pmatrix} \sigma_\alpha^2 & \rho\sigma_\alpha\sigma_\beta \\ \rho\sigma_\alpha\sigma_\beta & \sigma_\beta^2 \end{pmatrix} \right), \qquad j = 1, \ldots, J$$

- So we now have two second-level regression models:

$$E[\alpha_j] = \gamma_0^\alpha + \gamma_1^\alpha u_j$$

$$E[\beta_j] = \gamma_0^\beta + \gamma_1^\beta u_j$$

Group-Level Covariates

- This is really an interaction specification, since:

$$y_i = \alpha_{ij} + \beta_{ij} X_i + \epsilon_i$$

$$\alpha_j = \gamma_0^\alpha + \gamma_1^\alpha u_j + \eta_j^\alpha$$

$$\beta_j = \gamma_0^\beta + \gamma_1^\beta u_j + \eta_j^\beta$$

- Therefore:

$$y_i = \left[\gamma_0^\alpha + \gamma_1^\alpha u_j + \eta_j^\alpha\right] + \left[\gamma_0^\beta + \gamma_1^\beta u_j + \eta_j^\beta\right] X_i + \epsilon_i$$

$$= \gamma_0^\alpha + \gamma_1^\alpha u_j + \eta_j^\alpha + \gamma_0^\beta X_i + \gamma_1^\beta X_i u_j + \eta_j^\beta X_i + \epsilon_i$$

- For radon, define the following coefficient correspondence:

γ_0^α	`intercept`
γ_0^β	`x`
γ_1^α	`u.full`
γ_1^β	`x:u.full`

A Quick Look Again At Uranium

```
> u
 [1] -0.6890476 -0.8473129 -0.1134588 -0.5933525 -0.1428905  0.3870567  0.2716137  0.2775787
 [9] -0.3323155  0.0958646 -0.6082198  0.2736846 -0.7353201  0.3437812 -0.0598604 -0.5049960
[17]  0.3395603 -0.6333907 -0.0241452  0.2638555  0.1557123  0.2950250  0.4149137  0.2242070
[25]  0.1966106 -0.0965208  0.5035291 -0.4005970 -0.7518722 -0.6633476  0.3090203 -0.0533860
[33]  0.1097329 -0.0078034 -0.8818289  0.3110299 -0.6915964 -0.6817088  0.1944477  0.4449037
[41]  0.3947344  0.1496003  0.0137648  0.1658618  0.1404226  0.0239509 -0.2100595 -0.0932267
[49]  0.2609325  0.3988499  0.2480469  0.4054518  0.2652217  0.2431501 -0.2047304 -0.0740277
[57] -0.1632922  0.4786040  0.2661111  0.2811483 -0.4180535  0.3663223  0.3805780  0.1931461
[65]  0.5280249 -0.2120454  0.0631156 -0.6834365  0.2372121 -0.4746737  0.1163954  0.2698057
[73]  0.4707783  0.3160290 -0.0468401  0.4975945  0.1500824 -0.6720297  0.2124142 -0.1474843
[81]  0.1832378  0.2360361  0.4632119 -0.0900243  0.3552870

> u.full
  [1] -0.6890476 -0.6890476 -0.6890476 -0.6890476 -0.8473129 -0.8473129 -0.8473129 -0.8473129
  [9] -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129
 [17] -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129
 [25] -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129
 [33] -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129
 [41] -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129 -0.8473129
   :
[873] -0.1474843 -0.1474843 -0.1474843 -0.1474843 -0.1474843 -0.1474843 -0.1474843 -0.1474843
[881] -0.1474843 -0.1474843 -0.1474843 -0.1474843 -0.1474843 -0.1474843 -0.1474843  0.1832378
[889]  0.1832378  0.1832378  0.2360361  0.4632119  0.4632119  0.4632119  0.4632119  0.4632119
[897]  0.4632119  0.4632119  0.4632119  0.4632119  0.4632119  0.4632119  0.4632119  0.4632119
[905] -0.0900243 -0.0900243 -0.0900243 -0.0900243 -0.0900243 -0.0900243 -0.0900243 -0.0900243
[913] -0.0900243 -0.0900243 -0.0900243 -0.0900243 -0.0900243  0.3552870  0.3552870
```

Group-Level Covariates

- So it is easiest to specify as:

```
M4 <- glmer(y ~ x + u.full + x:u.full + (1 + x | county))
summary(M4)
Random effects:
Groups   Name        Variance Std.Dev. Corr
 county   (Intercept) 0.0156   0.125
 x           0.0941   0.307    0.409    0.409
 Residual             0.5617   0.749
number of obs: 919, groups: county, 85

Fixed effects:
            Estimate Std. Error t value      Correlation of Fixed Effects:
(Intercept)   1.4686     0.0353    41.6              (Intr) x      u.full
x            -0.6710     0.0844    -7.9      x        -0.241
u.full        0.8081     0.0907     8.9      u.full    0.207 -0.092
x:u.full     -0.4195     0.2271    -1.8      x:u.full -0.093  0.173 -0.231
```

Putting Pieces Together

- The `coef(M4)` command returns:

```
   (Intercept)        x  u.full x:u.full
1       1.4586 -0.64699 0.80806 -0.41946
2       1.4958 -0.88908 0.80806 -0.41946
3       1.4770 -0.64671 0.80806 -0.41946
:
85      1.4388 -0.70110 0.80806 -0.41946
```

- The first two columns vary by group from:

```
(1 + x | county)
```

- The second two columns do not vary by group because:

```
y ~ x + u.full + x:u.full
```

- Recall that `x` is not a group-level covariate, but is specified at both levels so that β is allowed to vary at the second level.

Putting Pieces Together

- We can collect the standard error coefficient for α as follows:

```
(a.se.M4 <- se.coef(M4)$county[,1])
      1       2       3       4       5
0.115104 0.088738 0.113913 0.107827 0.114849
:
     81      82      83      84      85
0.113973 0.123016 0.102748 0.104667 0.121417
```

- And the standard error coefficient for β is:

```
(b.se.M4 <- se.coef(M4)$county[,2])
     1      2      3      4      5
0.27765 0.24657 0.25677 0.23350 0.27673
:
    81     82     83     84     85
0.25710 0.30611 0.24339 0.27438 0.30546
```

Putting Pieces Together

- We obtain the estimate J and separate estimates of $\hat{\alpha}_j = \gamma_0^\alpha + \gamma_1^\alpha u_j$ from:

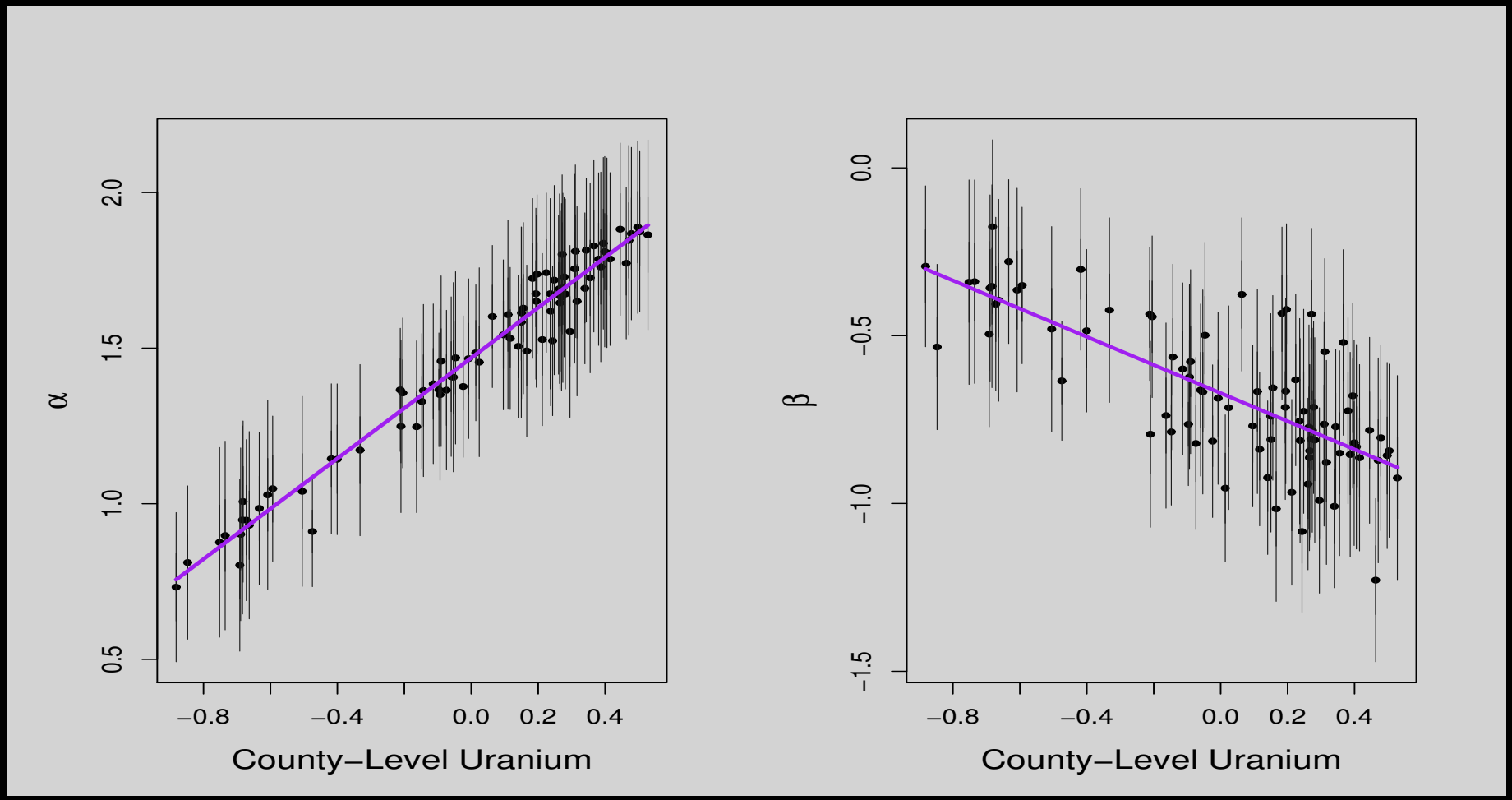
```
(a.hat.M4 <- coef(M4)$county[,1] + coef(M4)$county[,3]*u)
 [1] 0.90183 0.81112 1.38532 1.04836 1.36381
 [6] 1.76082 1.80116 1.72896 1.17220 1.54278
[11] 1.02871 1.69382 0.89785 1.81452 1.40840
[16] 1.03980 1.69203 0.98504 1.37662 1.69093
[21] 1.62833 1.55381 1.78631 1.74244 1.73766
[26] 1.36610 1.87382 1.14298 0.87602 0.93101
[31] 1.75509 1.40663 1.60761 1.46626 0.73204
[36] 1.81111 0.80251 1.00689 1.65030 1.88213
[41] 1.83688 1.58336 1.48489 1.49105 1.50566
[46] 1.45475 1.24872 1.35052 1.67190 1.80985
[51] 1.71852 1.80635 1.64544 1.52360 1.35592
[56] 1.36506 1.24744 1.86792 1.68286 1.67397
[61] 1.14452 1.82848 1.78606 1.67482 1.86403
[66] 1.36588 1.60164 0.94786 1.61865 0.91082
[71] 1.53118 1.66351 1.84605 1.65053 1.46872
[76] 1.88869 1.61307 0.94731 1.52746 1.32857
[81] 1.72395 1.67474 1.77278 1.45802 1.72586
```

Putting Pieces Together

- We can also get the estimate J separate estimates of $\hat{\beta}_j = \gamma_0^\beta + \gamma_1^\beta u_j$ from

```
(b.hat.M4 <- coef(M4)$county[,2] + coef(M4)$county[,4]*u)
 [1] -0.35796 -0.53366 -0.59912 -0.35033 -0.56350
 [6] -0.85411 -0.43575 -0.71334 -0.42378 -0.76886
[11] -0.36401 -0.78178 -0.33904 -0.77142 -0.66199
[16] -0.48011 -1.00850 -0.27916 -0.81425 -0.77257
[21] -0.65545 -0.99096 -0.86368 -0.63147 -0.42185
[26] -0.76403 -0.84279 -0.48522 -0.34061 -0.39440
[31] -0.76366 -0.66763 -0.66642 -0.68630 -0.29324
[36] -0.54770 -0.49519 -0.17535 -0.66512 -0.78201
[41] -0.67906 -0.73999 -0.95435 -1.01596 -0.92309
[46] -0.71455 -0.79364 -0.62319 -0.94192 -0.81929
[51] -0.72530 -0.83096 -0.86356 -1.08360 -0.44362
[56] -0.82123 -0.73802 -0.80408 -0.84303 -0.81096
[61] -0.30246 -0.52019 -0.72368 -0.71358 -0.92401
[66] -0.43552 -0.37705 -0.35267 -0.81248 -0.63459
[71] -0.83835 -0.80751 -0.87154 -0.87757 -0.49857
[76] -0.85706 -0.80950 -0.40586 -0.96665 -0.78664
[81] -0.43324 -0.75457 -1.22833 -0.57739 -0.85013
```

Figure 13.2 from Gelman & Hill



Graphing Commands

```
postscript("Class.Multilevel/figure13.2.ps")

lower <- a.hat.M4 - a.se.M4
upper <- a.hat.M4 + a.se.M4
par(mfrow=c(1,2),mar=c(5,5,5,3),bg="lightgray",cex.lab=1.4,cex.axis=1.1)
plot(u, a.hat.M4, ylim=range(lower,upper), pch=20,
     xlab="County-Level Uranium", ylab=expression(alpha))
segments (u, lower, u, upper, lwd=.5, col="gray10")
curve(fixef(M4)[1] + fixef(M4)[3]*x, lwd=3, col="purple", add=TRUE)

lower <- b.hat.M4 - b.se.M4
upper <- b.hat.M4 + b.se.M4
plot (u, b.hat.M4, ylim=range(lower,upper), pch=20,
      xlab="County-Level Uranium", ylab=expression(beta))
segments (u, lower, u, upper, lwd=.5, col="gray10")
curve(fixef(M4)[2] + fixef(M4)[4]*x, lwd=3, col="purple", add=TRUE)

dev.off()
```

What About Adding Uranium at the Both Levels?

```
M5 <- glmer(y ~ x + u.full + (1 + x + u.full| county))
summary(M5)
Random effects:
 Groups   Name        Variance Std.Dev. Corr
 county   (Intercept) 5.70e-11 7.55e-06
          x           1.47e-01 3.83e-01 0.000
          u.full      1.27e-01 3.56e-01 0.000 0.661
 Residual             5.61e-01 7.49e-01
Number of obs: 919, groups: county, 85

Fixed effects:                              Correlation of Fixed Effects:
            Estimate Std. Error t value                    (Intr) x
(Intercept)   1.4586     0.0319    45.8           x       -0.336
x            -0.6379     0.0888    -7.2           u.full   0.086  0.143
u.full        0.7616     0.0976     7.8
```

It doesn't really work since it is now just a substitute for county.

Panel Data as Group Membership

- 2,000 Australian adolescents with smoking measured every 6 months for 3 years.
- So observations are nested (grouped) with persons.
- Specified model for case j at wave t:

  $$p(y_i = 1) = \text{logit}^{-1}(\beta_0 + \beta_1 \text{psmoke}_{ij} + \beta_2 \text{female}_{ij} + \beta_3(1 - \text{female}_{ij})t[i] + \beta_4(\text{female}_{ij})t[i] + \alpha_j)$$

  for a representation that has a matrix for individual effects that don't change ($[i]$) and another for group effects that do not change (ij).
- So the jth person has multiple waves on the jth row of the $\mathbf{X}$ matrix.
- Software note: for logit/probit link functions `glmer` sets $\sigma_y = 1$ for identifiability.

Data Load

```
lapply(c("lme4","arm"),library, character.only=TRUE)
smoking <- read.table("http://jeffgill.org/files/jeffgill/files/smoke_pub.dat_.txt",
    header=TRUE)
smoking[c(1:8,(nrow(smoking)-7):nrow(smoking)),]
1        1        1      0    1      0
2        1        1      0    2      0
3        1        1      0    4      0
4        1        1      0    5      0
5        1        1      0    6      0
6        2        0      0    1      0
7        2        0      0    2      0
8724  1758        1      1    4      0
8725  1758        1      1    5      0
8726  1758        1      1    6      0
8727  1759        1      1    4      0
8728  1759        1      1    5      0
8729  1759        1      1    6      0
8730  1760        0      0    5      0
```

Data Exploration

```
summary(smoking)
     newid          sex.1.F.         parsmk          wave          smkreg
 Min.   :   1   Min.   :0.000   Min.   :0.00   Min.   :1.00   Min.   :0.000
 1st Qu.: 417   1st Qu.:0.000   1st Qu.:0.00   1st Qu.:2.00   1st Qu.:0.000
 Median : 830   Median :1.000   Median :0.00   Median :4.00   Median :0.000
 Mean   : 847   Mean   :0.542   Mean   :0.35   Mean   :3.68   Mean   :0.125
 3rd Qu.:1280   3rd Qu.:1.000   3rd Qu.:1.00   3rd Qu.:5.00   3rd Qu.:0.000
 Max.   :1760   Max.   :1.000   Max.   :1.00   Max.   :6.00   Max.   :1.000
table(smoking$wave)
  1    2    3    4    5    6
876 1571 1601 1587 1570 1525
cor(smoking)
            newid sex.1.F.     parsmk       wave   smkreg
newid    1.000000 0.085630  0.0229614  0.1660314 0.040525
sex.1.F. 0.085630 1.000000  0.0149581  0.0143264 0.045383
parsmk   0.022961 0.014958  1.0000000 -0.0072693 0.153724
wave     0.166031 0.014326 -0.0072693  1.0000000 0.076927
smkreg   0.040525 0.045383  0.1537244  0.0769268 1.000000
```

Varying Intercept Logit Multilevel Model, *No Group-Level Explanatory Variables*

```
lmer.out <- glmer(smkreg ~ wave + (1|newid), data=smoking,
                  family=binomial(link=logit))
display(lmer.out)
glmer(formula = smkreg ~ wave + (1 | newid), data = smoking,
    family = binomial(link = logit))
            coef.est coef.se
(Intercept) -6.41     0.27
wave         0.21     0.05

Error terms:
 Groups   Name        Std.Dev.
 newid    (Intercept) 4.17
 Residual             1.00
number of obs: 8730, groups: newid, 1760
AIC = 17774.7, DIC = 17769
deviance = 17768.7
```

Varying Intercept Logit Multilevel Model, *Adding Group-Level Explanatory Variables*

```
lmer.out <- glmer(smkreg ~ wave + sex.1.F. + parsmk + (1|newid), data=smoking,
                  family=binomial(link=logit))
display(lmer.out)
glmer(formula = smkreg ~ wave + sex.1.F. + parsmk + (1 | newid),
    data = smoking, family = binomial(link = logit))
            coef.est coef.se
(Intercept) -6.60     0.30
wave         0.25     0.04
sex.1.F.     0.05     0.32
parsmk       1.17     0.31

Error terms:
 Groups   Name        Std.Dev.
 newid    (Intercept) 4.18
 Residual             1.00
number of obs: 8730, groups: newid, 1760
AIC = 3935.8, DIC = 3925.8
deviance = 3925.8
```

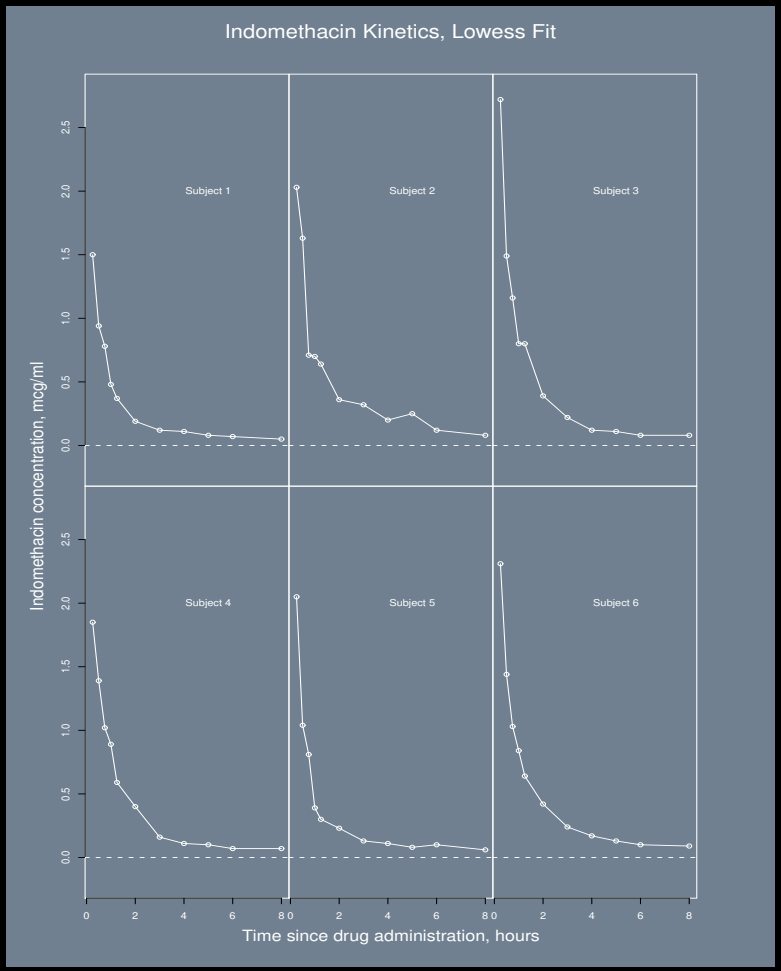
Varying-Intercept, Varying-Slope Logit Multilevel Model, *Drop Parents Smoking, Add an Interaction*

```
lmer.out <- glmer(smkreg ~ wave + sex.1.F. + wave:sex.1.F. + (1|newid), data=smoking,
                  family=binomial(link=logit))
display(lmer.out)
glmer(formula = smkreg ~ wave + sex.1.F. + wave:sex.1.F. + (1 |
    newid), data = smoking, family = binomial(link = logit))
             coef.est coef.se
(Intercept)   -4.85     0.26
wave           0.10     0.05
sex.1.F.      -0.88     0.36
wave:sex.1.F.  0.21     0.07

Error terms:
 Groups   Name        Std.Dev.
 newid    (Intercept) 3.47
 Residual             1.00
number of obs: 8730, groups: newid, 1760
AIC = 4086.4, DIC = 4076.4
deviance = 4076.4
```

Nonlinear Random Effects Example for Indomethacin Trials

- A pharmacokinetic analysis of the nonsteroidal anti-inflammatory drug Indomethacin from Kwan, Breault, Umbenhauer, McMahon, and Duggan, (1976) *Journal of Pharmacokinetics and Biopharmaceutics*, 4, 255-280.
- 6 volunteers received bolus intravenous injections of the same dose of Indomethacin.
- Plasma concentration (in mcg/ml) is measured 11 times between 15 minutes and 8 hours postinjection.
- Motivation for multilevel analysis: all subjects show a characteristic decay curve, but rates differ.



Nonlinear Random Effects Example for Indomethacin Trials

```
"Indometh" <-
  structure(list(
   Subject = structure(ordered(c(1, 1, 1, 1, 1, 1,
     1, 1, 1, 1, 1, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 6, 6, 6, 6, 6,
     6, 6, 6, 6, 6, 6, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 4, 4, 4, 4,
     4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5), levels=1:6),
     class = c("ordered", "factor"),
     .Label = c("1", "4", "2", "5", "6", "3")),
   time = c(0.25, 0.5, 0.75, 1, 1.25, 2, 3, 4, 5, 6, 8, 0.25, 0.5,
     0.75, 1, 1.25, 2, 3, 4, 5, 6, 8, 0.25, 0.5, 0.75, 1, 1.25, 2, 3,
     4, 5, 6, 8, 0.25, 0.5, 0.75, 1, 1.25, 2, 3, 4, 5, 6, 8, 0.25,
     0.5, 0.75, 1, 1.25, 2, 3, 4, 5, 6, 8, 0.25, 0.5, 0.75, 1, 1.25,
     2, 3, 4, 5, 6, 8),
   conc = c(1.5, 0.94, 0.78, 0.48, 0.37, 0.19, 0.12, 0.11,
     0.08, 0.07, 0.05, 2.03, 1.63, 0.71, 0.7, 0.64, 0.36, 0.32, 0.2,
     0.25, 0.12, 0.08, 2.72, 1.49, 1.16, 0.8, 0.8, 0.39, 0.22, 0.12,
     0.11, 0.08, 0.08, 1.85, 1.39, 1.02, 0.89, 0.59, 0.4, 0.16, 0.11,
     0.1, 0.07, 0.07, 2.05, 1.04, 0.81, 0.39, 0.3, 0.23, 0.13, 0.11,
     0.08, 0.1, 0.06, 2.31, 1.44, 1.03, 0.84, 0.64, 0.42, 0.24, 0.17,
     0.13, 0.1, 0.09)),
row.names = 1:66,
class = c("nfnGroupedData", "nfGroupedData", "groupedData", "data.frame"),
formula = conc ~ time | Subject,
labels = list(X = "Time since drug administration",
  y = "Indomethacin concentration"),
units = list(X = "(hr)", y = "(mcg/ml)"))
```

Nonlinear Random Effects Example for Indomethacin Trials

```
Indometh

Grouped Data: conc ~ time | Subject
   Subject time conc
1        1 0.25 1.50
2        1 0.50 0.94
:
65       6 6.00 0.10
66       6 8.00 0.09

names(Indometh)
"Subject" "time"    "conc"

class(Indometh)
"nfnGroupedData" "nfGroupedData" "groupedData" "data.frame"

formula(Indometh)
Subject ~ time + conc
```

Nonlinear Random Effects Example for Indomethacin Trials

- Biexponential model:

$$y_{ij} = \phi_1 \exp(-\exp(\phi_2)t_j) + \phi_3 \exp(-\exp(\phi_4)t_j) + \epsilon_{ij},$$

with: individual i at time t_j, $e_{ij} \sim N(0, \sigma^2)$, $\phi_2 > \phi_4$ for identifiability.

Nonlinear Random Effects Example for Indomethacin Trials

- Biexponential model:

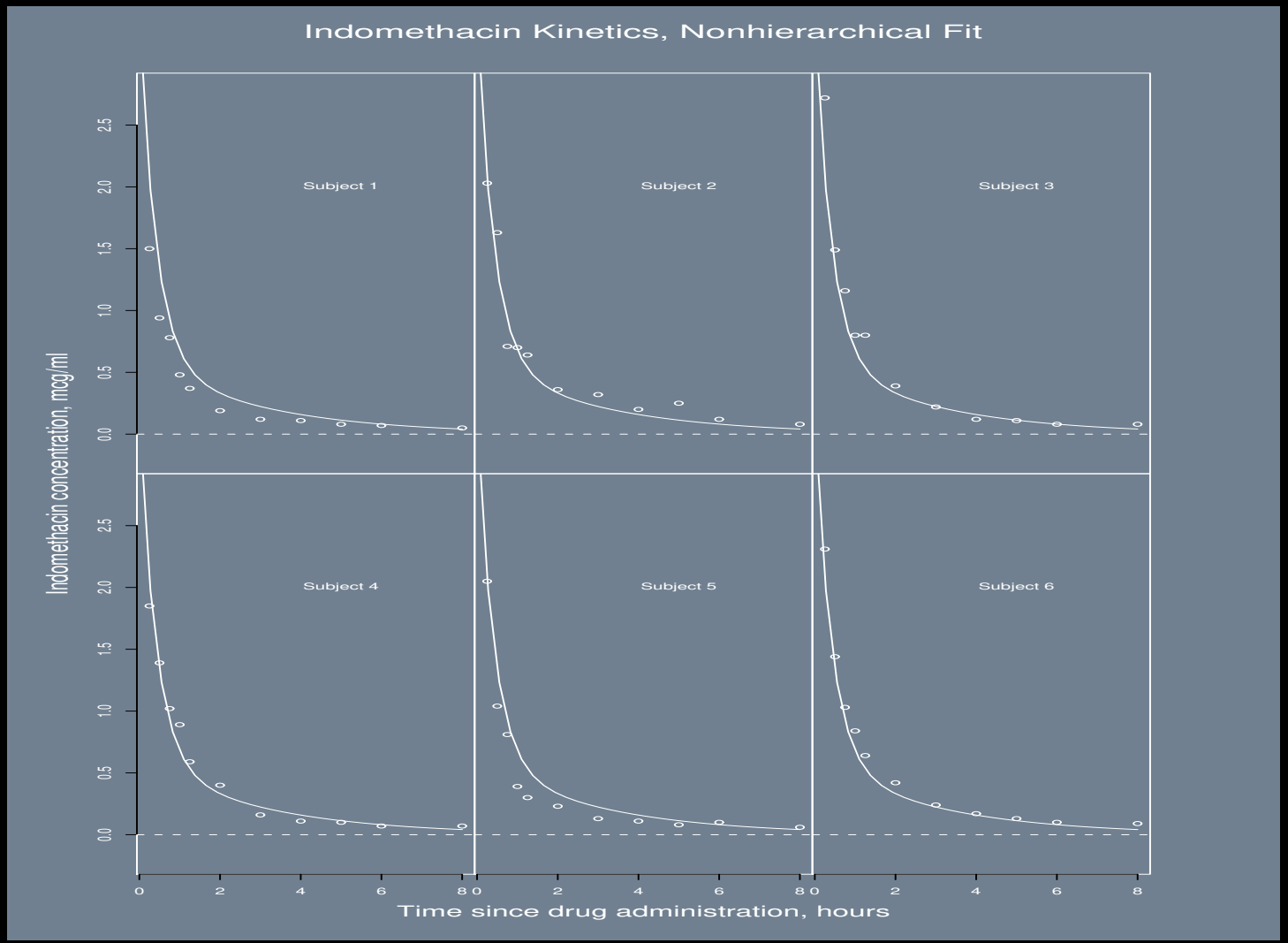
$$y_{ij} = \phi_1 \exp(-\exp(\phi_2)t_j) + \phi_3 \exp(-\exp(\phi_4)t_j) + \epsilon_{ij},$$

  with: individual i at time t_j, $e_{ij} \sim N(0, \sigma^2)$, $\phi_2 > \phi_4$ for identifiability.

- First fit a model ignoring hierarchy, getting fixed effect estimates (*full pooling*):

```
library(nlme)
indo.pop.nls <- nls(conc ~ SSbiexp(time, A1, lrc1, A2, lrc2), data = Indometh)
summary(indo.pop.nls)
Parameters:
     Estimate Std. Error t value Pr(>|t|)
A1      2.773      0.253   10.95    4e-16
lrc1    0.886      0.222    3.99  0.00018
A2      0.607      0.267    2.27  0.02660
lrc2   -1.092      0.409   -2.67  0.00966
---
Residual standard error: 0.1745 on 62 degrees of freedom
```

Nonlinear Random Effects Example for Indomethacin Trials



Nonlinear Random Effects Example for Indomethacin Trials

- Now fit a separate model to each individual subject ignoring population effects (*no pooling*).

```
indo.term.lis <- nlsList(conc ~ SSbiexp(time, A1, lrc1, A2, lrc2), data = Indometh)
summary(indo.term.lis)

Coefficients:
   A1
  Estimate Std. Error   t value     Pr(>|t|)
1 2.029277  0.2023875 10.026695 3.388285e-07
2 2.827673  0.2311604 12.232518 3.536510e-04
3 5.468312  1.8759966  2.914884 1.087070e-02
4 2.198132  0.3155032  6.967066 7.942588e-06
5 3.566103  0.3245732 10.987053 5.630248e-06
6 3.002250  0.3503106  8.570251 3.069467e-07
   lrc1
   Estimate Std. Error  t value     Pr(>|t|)
1 0.5793887  0.2295508 2.524011 2.347391e-03
2 0.8013195  0.1803742 4.442540 5.193756e-02
3 1.7497936  0.3108862 5.628406 2.937754e-04
4 0.2423124  0.2427792 0.998077 1.402737e-01
```

```
5 1.0407660  0.1636874 6.358253 1.986106e-04
6 1.0882119  0.2564197 4.243870 3.504364e-05
   A2
   Estimate Std. Error   t value     Pr(>|t|)
1 0.1915474  0.2037201 0.9402482 0.1269760002
2 0.4989175  0.1822390 2.7377104 0.1927034117
3 1.6757521  0.2814723 5.9535238 0.0002075745
4 0.2545223  0.3716832 0.6847828 0.2914158655
5 0.2914970  0.1592207 1.8307727 0.0811792830
6 0.9685230  0.2905245 3.3337056 0.0001646898

   lrc2
    Estimate Std. Error   t value     Pr(>|t|)
1 -1.7877849  1.4495070 -1.233374 0.0573694854
2 -1.6353512  0.4779239 -3.421781 0.1146482023
3 -0.4122004  0.1680153 -2.453351 0.0232031740
4 -1.6026860  1.4786607 -1.083877 0.1138959965
5 -1.5068522  0.7133811 -2.112268 0.0511506022
6 -0.8731358  0.2715939 -3.214858 0.0002066297

Residual standard error: 0.0755502 on 42 degrees of freedom # 66-24
```

Nonlinear Random Effects Example for Indomethacin Trials

- Finally we get a fully mixed effects biexponential model:

$$y_{ij} = (\beta_1 - b_{1i}\exp(-\exp(\beta_2 - b_{2i})t_j) + (\beta_3 - b_{3i}\exp(-\exp(\beta_4 - b_{4i})t_j + \epsilon_{ij}$$

where the β's give the estimated mean population effects, and the b_i's give the individual deviations, with assumed mean zero.

Nonlinear Random Effects Example for Indomethacin Trials

- Finally we get a fully mixed effects biexponential model:

$$y_{ij} = (\beta_1 - b_{1i}\exp(-\exp(\beta_2 - b_{2i})t_j) + (\beta_3 - b_{3i}\exp(-\exp(\beta_4 - b_{4i})t_j + \epsilon_{ij}$$

  where the β's give the estimated mean population effects, and the b_i's give the individual deviations, with assumed mean zero.

- Also we assume no covariances by using only the diagonal for computation.

```
indo.nlme <- nlme( indo.term.lis, random = pdDiag(A1 + lrc1 + A2 + lrc2 ~ 1) )

summary(indo.nlme)

Data: Indometh
       AIC       BIC   logLik
  -91.18562 -71.47873 54.59281
```

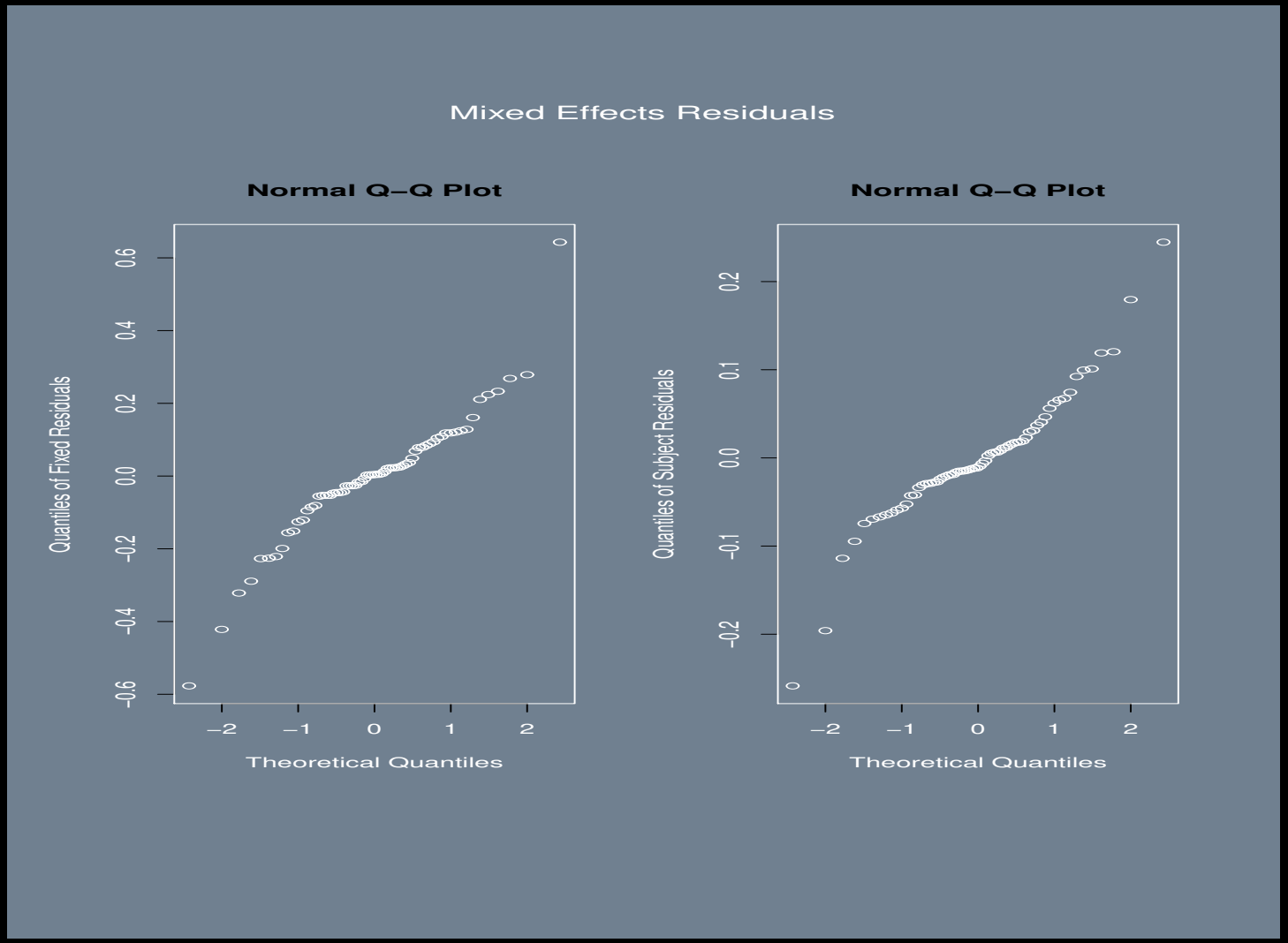
```
Random effects:
 Formula: list(A1 ~ 1, lrc1 ~ 1, A2 ~ 1, lrc2 ~ 1)
 Level: Subject
 Structure: Diagonal
             A1    lrc1      A2       lrc2 Residual
StdDev: 0.57141 0.15808 0.11160 8.2778e-06 0.081493

Fixed effects: list(A1 ~ 1, lrc1 ~ 1, A2 ~ 1, lrc2 ~ 1)
        Value Std.Error DF t-value p-value
A1    2.82754   0.26401 57 10.7099   0e+00
lrc1  0.77362   0.11003 57  7.0313   0e+00
A2    0.46147   0.11281 57  4.0908   1e-04
lrc2 -1.34410   0.23108 57 -5.8167   0e+00

Standardized Within-Group Residuals:
     Min       Q1      Med       Q3      Max
-3.17338 -0.35627 -0.12853  0.34232  3.00251

Number of Observations: 66
Number of Groups: 6
```

Nonlinear Random Effects Example for Indomethacin Trials



Nonlinear Random Effects Example for Indomethacin Trials

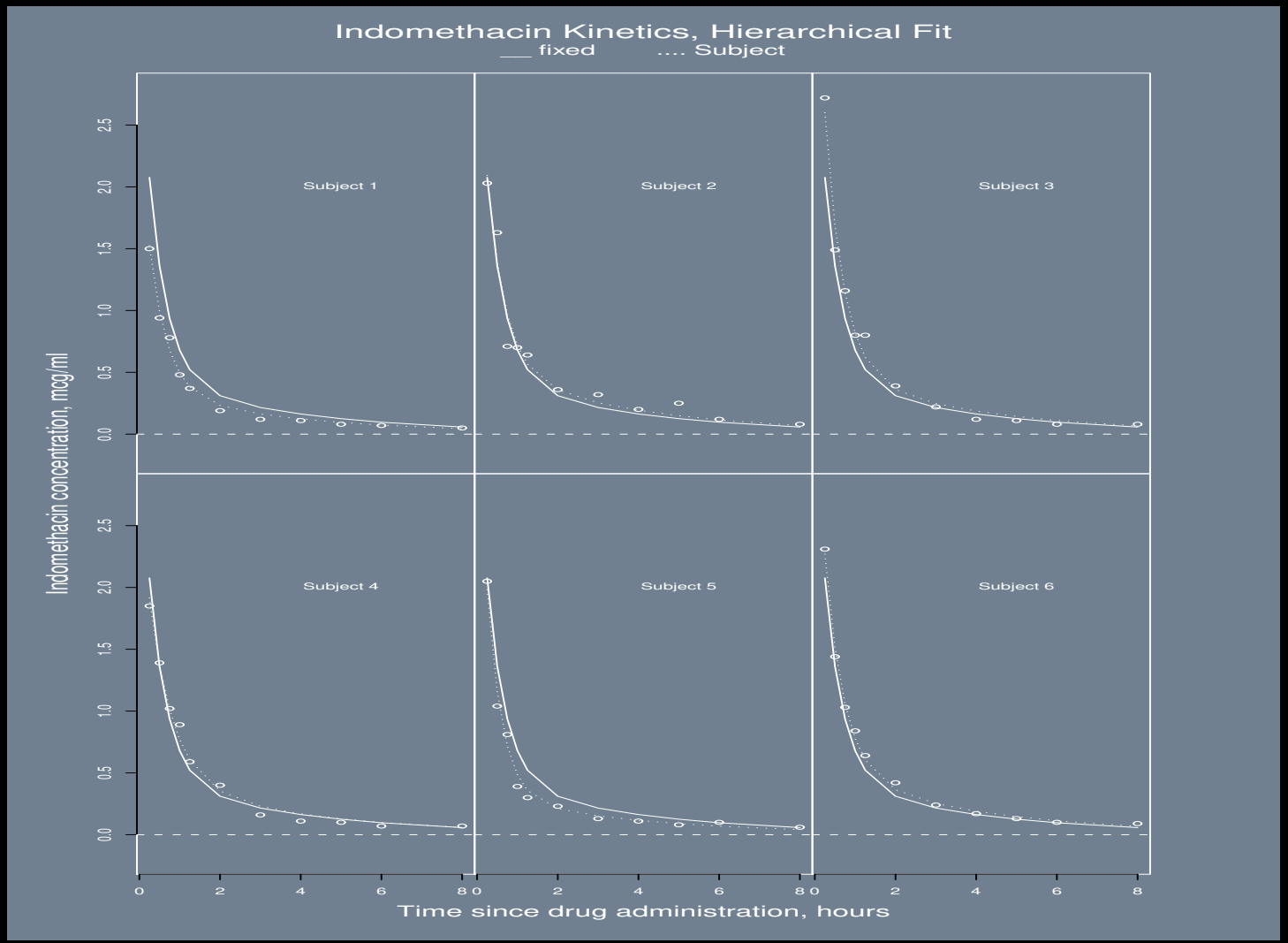
```
( be.fixed.terms <- as.vector(indo.nlme$coefficients$fixed) )

[1]  2.8275982  0.7732937  0.4610679 -1.3449124

( be.random.terms <- as.matrix(indo.nlme$coefficients$random$Subject) )

           A1          lrc1          A2           lrc2
1 -0.73966229  0.027813190 -0.11124141  6.553013e-17
2 -0.07054535  0.028667640  0.08711800 -1.855721e-16
3  0.80060558  0.004134323  0.06711258  4.906331e-17
4 -0.56554287 -0.231256957  0.01169103  7.287592e-17
5  0.41273288  0.198153725 -0.13174119  3.397749e-17
6  0.16241204 -0.027511921  0.07706099 -3.587477e-17
```

Nonlinear Random Effects Example for Indomethacin Trials

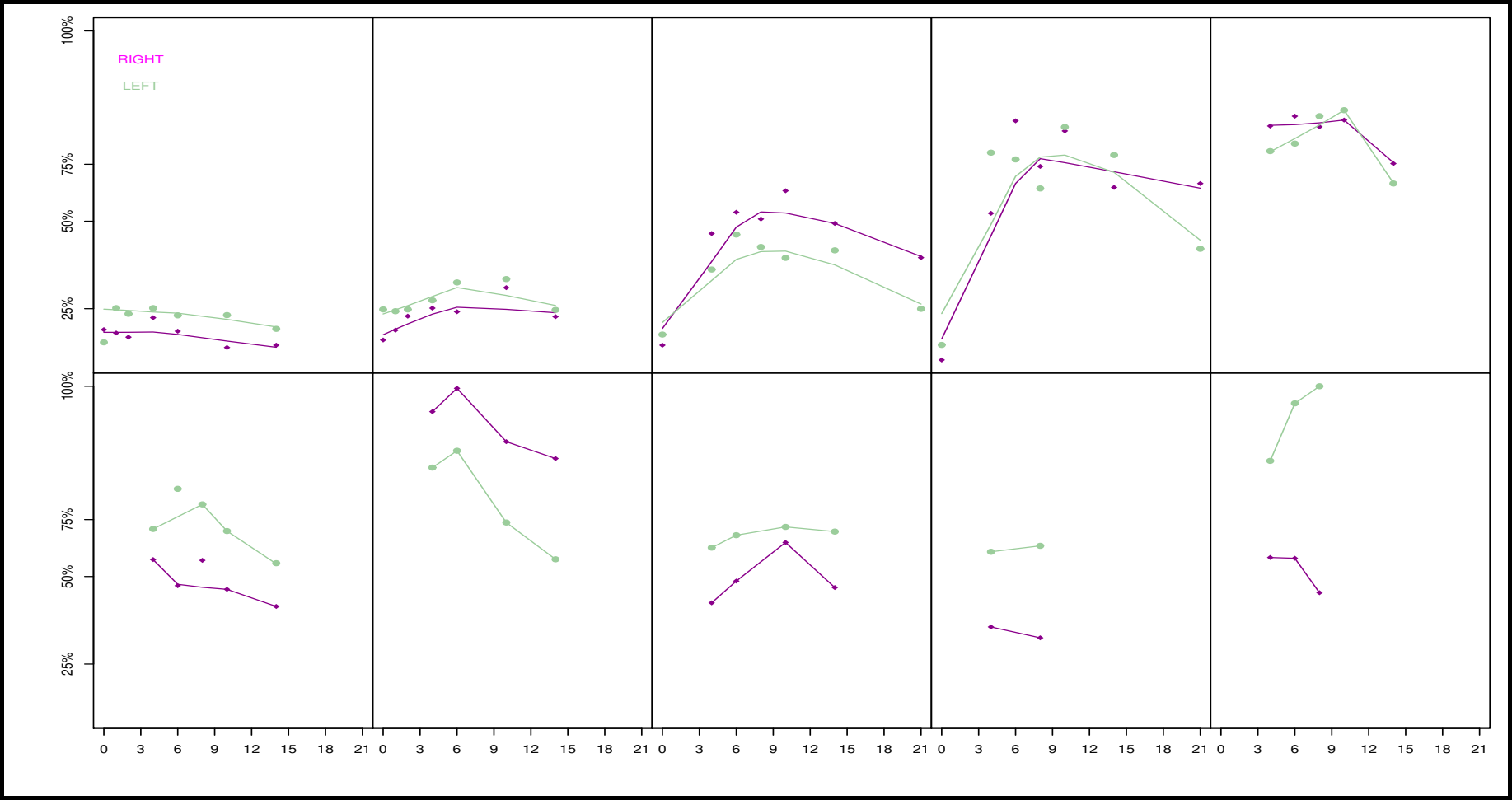


Nested Classification Factors

- Experiment recording mean pixel intensity of the right and left lymph-nodes in the axillary region from (tomography) CT scans of 10 dogs over 14 days after intravenous application of a dye contrast.
- Data:

```
connect1 <- url("http://jeffgill.org/data/Pixel.rda")
load(connect1);   close(connect1)
Pixel[,"Side"] <- as.numeric( Pixel[,"Side"] )
Pixel[1:10,]
   Dog Side day  pixel
1    1    2   0 1045.8
2    1    2   1 1044.5
3    1    2   2 1042.9
4    1    2   4 1050.4
5    1    2   6 1045.2
6    1    2  10 1038.9
7    1    2  14 1039.8
8    2    2   0 1041.8
9    2    2   1 1045.6
10   2    2   2 1051.0
```

Nested Classification Factors



Nested Classification Factors

```
postscript("Class.Multilevel/dogs1.ps")
J = max(as.numeric(Pixel$Dog))
day.range <- c(min(as.numeric(Pixel$day)), max(as.numeric(Pixel$day)))
pixel.range <- c(min(as.numeric(Pixel$pixel)), max(as.numeric(Pixel$pixel)))
par(oma=c(5,5,1,1),mar=c(0,0,0,0),mfrow=c(J/(J/2),J/2),bg="white")
for (i in 1:J)  {
    dog.i <- Pixel[Pixel["Dog"]==i,]
    plot(dog.i[dog.i["Side"]==2,][,3:4], xlim=day.range, ylim=pixel.range,
        pch=18,col="darkmagenta",yaxt="n",xaxt="n")
    lines(lowess(dog.i[dog.i["Side"]==2,][,3:4], f=0.9), col="darkmagenta")
    points(dog.i[dog.i["Side"]==1,][,3:4], pch=19,col="darkseagreen3")
    lines(lowess(dog.i[dog.i["Side"]==1,][,3:4], f=0.9), col="darkseagreen3")
    if (i > 5) axis(side=1,labels=seq(0,21,by=3),at=seq(0,21,by=3))
    if (i == 1 | i==6) axis(side=2,labels=names(quantile(Pixel$pixel)[-1]),
        at=quantile(Pixel$pixel)[-1])
    if (i == 1)  { text(3,1150,"RIGHT",col="magenta")
                   text(3,1140,"LEFT",col="darkseagreen3") }
}
dev.off()
```

Nested Classification Factors

- We want to account for nesting of sides with dogs.
- We also need to think carefully about where to put intercepts in the various levels (choices).
- Note that with nesting levels we can qualitatively different results for standard error terms.
- There is an obvious nonlinear effect to account for in days with a peak at about 10.
- Most general model:

$$
\begin{aligned}
y_{ijt} &= \beta_0 + \beta_1 T_i + \beta_2 T_i^2 + \beta_3 S_{ij} + \beta_{4,ij} + \beta_{5,ij} + \epsilon_{ijt} \\
\beta_4 &\sim N(\gamma_0 + \gamma_1 S_{ij} + \gamma_2 T_i, \sigma^2_{\beta_4,}) \\
\beta_5 &\sim N(\delta_0 + \delta_1 T_i, \sigma^2_{\beta_5,})
\end{aligned}
$$

where:

y_{ijt}	pixels for ith dog, side j, at time t
T_{ij}	ith dog's time t (not all measured at regular intervals!)
S_{ij}	ith dog's side: 1 (left) or 2 (right).

Nested Classification Factors

- Random intercepts model:

```
fm1Pixel <- glmer(pixel ~ day + I(day^2) + (1 | Dog), data = Pixel)
summary(fm1Pixel)
   AIC   BIC logLik deviance REMLdev
 895.2 908.4 -442.6    886.9   885.2
Random effects:
 Groups   Name        Variance Std.Dev.
 Dog      (Intercept) 633.89   25.177
 Residual             263.95   16.247
Number of obs: 102, groups: Dog, 10

Fixed effects:
             Estimate Std. Error t value
(Intercept) 1074.16253    9.12496  117.72
day            4.94201    1.03686    4.77
I(day^2)      -0.25047    0.05303   -4.72

Correlation of Fixed Effects:
            (Intr) day
day         -0.426
I(day^2)     0.355 -0.945
```

Nested Classification Factors

- Checking for a non-grouped difference between the left and right sides across dogs:

```
fm2Pixel <- glmer(pixel ~ day + I(day^2) + Side + (1 | Dog), data = Pixel)
summary(fm2Pixel)
   AIC BIC logLik deviance REMLdev
 890.2 906 -439.1      884   878.2
Random effects:
 Groups   Name        Variance Std.Dev.
 Dog      (Intercept) 634.54   25.190
 Residual             258.56   16.080
Number of obs: 102, groups: Dog, 10

Fixed effects:
             Estimate Std. Error t value
(Intercept) 1082.28413   10.28304  105.25
day            4.93811    1.02628    4.81
I(day^2)      -0.25029    0.05249   -4.77
Side          -5.40196    3.18425   -1.70

Correlation of Fixed Effects:
            (Intr) day    I(d^2)
day         -0.374
I(day^2)     0.311 -0.945
Side        -0.464  0.000  0.000
```

Nested Classification Factors

```
anova(fm2Pixel,fm1Pixel)

fm1Pixel: pixel ~ day + I(day^2) + (1 | Dog)
fm2Pixel: pixel ~ day + I(day^2) + Side + (1 | Dog)
         Df     AIC     BIC  logLik  Chisq Chi Df Pr(>Chisq)
fm1Pixel  5  896.87  910.00 -443.44
fm2Pixel  6  895.94  911.69 -441.97 2.9352      1    0.08667
```

Nested Classification Factors

- A model that nests the side in dogs as well as the days in dogs, days has intercept:

```
fm3Pixel <- glmer(pixel ~ day + I(day^2) + (Side | Dog) + (-1 + day | Dog),
                  data = Pixel); summary(fm3Pixel)

   AIC   BIC logLik deviance REMLdev
 843.4 864.4 -413.7    829.7   827.4
Random effects:
 Groups   Name        Variance  Std.Dev. Corr
 Dog      (Intercept) 1937.2268 44.014
          Side         565.3087 23.776   -0.746
 Dog      day            3.1791  1.783
 Residual                81.3617  9.020
Number of obs: 102, groups: Dog, 10

Fixed effects:
            Estimate Std. Error t value
(Intercept) 1073.6530     9.7331  110.31
day            6.2335     0.8655    7.20
I(day^2)      -0.3685     0.0340  -10.84

Correlation of Fixed Effects:
            (Intr) day
day         -0.202
I(day^2)     0.195 -0.679
```

Nested Classification Factors

```
anova( fm3Pixel, fm2Pixel )

fm2Pixel: pixel ~ day + I(day^2) + Side + (1 | Dog)
fm3Pixel: pixel ~ day + I(day^2) + (Side | Dog) + (-1 + day | Dog)
          Df     AIC     BIC  logLik Chisq Chi Df Pr(>Chisq)
fm2Pixel  6  895.94  911.69 -441.97
fm3Pixel  8  845.69  866.69 -414.84 54.25      2  1.658e-12
```

Nested Classification Factors

- Is it worth having Side as a fixed effect too?

```
fm4Pixel <- glmer(pixel ~ day + I(day^2) + Side + (Side | Dog) + (-1 + day | Dog),
                  data = Pixel); summary(fm4Pixel)
 AIC   BIC logLik deviance REMLdev
 838 861.7   -410    828.2     820
Random effects:
 Groups   Name        Variance  Std.Dev. Corr
 Dog      (Intercept) 1908.1242 43.6821
          Side         544.6817 23.3384  -0.741
 Dog      day            3.1794  1.7831
 Residual               81.2434  9.0135
Number of obs: 102, groups: Dog, 10

Fixed effects:
             Estimate Std. Error t value
(Intercept) 1086.41495   14.41456   75.37
day            6.23301    0.86512    7.20
I(day^2)      -0.36852    0.03398  -10.85
Side          -9.19399    7.62640   -1.21

Correlation of Fixed Effects:
            (Intr) day    I(d^2)
day         -0.136
I(day^2)     0.131 -0.679
Side        -0.737  0.000  0.000
```

Nested Classification Factors

```
anova( fm4Pixel, fm3Pixel )

fm3Pixel: pixel ~ day + I(day^2) + (Side | Dog) + (-1 + day | Dog)
fm4Pixel: pixel ~ day + I(day^2) + Side + (Side | Dog) + (-1 + day | Dog)
         Df     AIC     BIC  logLik  Chisq Chi Df Pr(>Chisq)
fm3Pixel  8  845.69  866.69 -414.84
fm4Pixel  9  846.20  869.83 -414.10 1.4834      1     0.2232
```