CHAPTER 12

Multilevel linear models: the basics

Multilevel modeling can be thought of in two equivalent ways:

- We can think of a generalization of linear regression, where intercepts, and possibly slopes, are allowed to vary by group. For example, starting with a regression model with one predictor, $y_i = \alpha + \beta x_i + \epsilon_i$, we can generalize to the varying-intercept model, $y_i = \alpha_{j[i]} + \beta x_i + \epsilon_i$, and the varying-intercept, varying-slope model, $y_i = \alpha_{j[i]} + \beta_{j[i]} x_i + \epsilon_i$ (see Figure 11.1 on page 238).
- Equivalently, we can think of multilevel modeling as a regression that includes a categorical input variable representing group membership. From this perspective, the group index is a factor with J levels, corresponding to J predictors in the regression model (or $2J$ if they are interacted with a predictor x in a varying-intercept, varying-slope model; or $3J$ if they are interacted with two predictors $X_{(1)}, X_{(2)}$; and so forth).

In either case, $J-1$ linear predictors are added to the model (or, to put it another way, the constant term in the regression is replaced by J separate intercept terms). The crucial multilevel modeling step is that these J coefficients are then themselves given a model (most simply, a common distribution for the J parameters α_j or, more generally, a regression model for the α_j's given group-level predictors). The group-level model is estimated simultaneously with the data-level regression of y.

This chapter introduces multilevel linear regression step by step. We begin in Section 12.2 by characterizing multilevel modeling as a compromise between two extremes: *complete pooling*, in which the group indicators are not included in the model, and *no pooling*, in which separate models are fit within each group. After laying out some notational difficulties in Section 12.5, we discuss in Section 12.6 the different roles of the individual- and group-level regressions. Chapter 13 continues with more complex multilevel structures.

12.1 Notation

We briefly review the notation for classical regression and then outline how it can be generalized for multilevel models. As we illustrate in the examples, however, no single notation is appropriate for all problems. We use the following notation for classical regression:

- Units $i = 1, \ldots, n$. By *units*, we mean the smallest items of measurement.
- Outcome measurements $y = (y_1, \ldots, y_n)$. These are the unit-level data being modeled.
- Regression predictors are represented by an $n \times k$ matrix X, so that the vector of predicted values is $\hat{y} = X\beta$, where $\hat{y}$ and β are column vectors of length n and k, respectively. We include in X the constant term (unless it is explicitly excluded from the model), so that the first column of X is all 1's. We usually label the coefficients as $\beta_0, \ldots, \beta_{k-1}$, but sometimes we index from 1 to k.
- For each individual unit i, we denote its row vector of predictors as X_i. Thus, $\hat{y}_i = X_i\beta$ is the prediction for unit i.

- For each predictor κ, we label the $(\kappa+1)^{st}$ column of X as $X_{(\kappa)}$ (assuming that $X_{(0)}$ is a column of 1's).
- Any information contained in the unit labels i should be coded in the regression inputs. For example, if $i = 1, \ldots, n$ represents the order in which persons i enrolled in a study, we should create a time variable t_i and, for example, include it in the matrix X of regression predictors. Or, more generally, consider transformations and interactions of this new input variable.

For multilevel models, we label:

- Groups $j = 1, \ldots, J$. This works for a single level of grouping (for example, students within schools, or persons within states).
- We occasionally use $k = 1, \ldots, K$ for a second level of grouping (for example, students within schools within districts; or, for a non-nested example, test responses that can be characterized by person or by item). In any particular example, we have to distinguish this k from the number of predictors in X. For more complicated examples we develop idiosyncratic notation as appropriate.
- Index variables $j[i]$ code group membership. For example, if $j[35] = 4$, then the 35^{th} unit in the data ($i = 35$) belongs to group 4.
- Coefficients are sometimes written as a vector β, sometimes as α, β (as in Figure 11.1 on page 238), with group-level regression coefficients typically called γ.
- We make our R and Bugs code more readable by typing α, β, γ as `a,b,g`.
- We write the varying-intercept model with one additional predictor as $y_i = \alpha_{j[i]} + \beta x_i + \epsilon_i$ or $y_i \sim \mathrm{N}(\alpha_{j[i]} + \beta x_i, \sigma_y^2)$. Similarly, the varying-intercept, varying-slope model is $y_i = \alpha_{j[i]} + \beta_{j[i]} x_i + \epsilon_i$ or $y_i \sim \mathrm{N}(\alpha_{j[i]} + \beta_{j[i]} x_i, \sigma_y^2)$.
- With multiple predictors, we write $y_i = X_i B + \epsilon_i$, or $y_i \sim \mathrm{N}(X_i B, \sigma_y^2)$. B is a matrix of coefficients that can be modeled using a general varying-intercept, varying-slope model (as discussed in the next chapter).
- Standard deviation is σ_y for data-level errors and $\sigma_\alpha, \sigma_\beta$, and so forth, for group-level errors.
- Group-level predictors are represented by a matrix U with J rows, for example, in the group-level model, $\alpha_j \sim \mathrm{N}(U_j \gamma, \sigma_\alpha^2)$. When there is a single group-level predictor, we label it as lowercase u.

12.2 Partial pooling with no predictors

As noted in Section 1.3, multilevel regression can be thought of as a method for compromising between the two extremes of excluding a categorical predictor from a model (*complete pooling*), or estimating separate models within each level of the categorical predictor (*no pooling*).

Complete-pooling and no-pooling estimates of county radon levels

We illustrate with the home radon example, which we introduced in Section 1.2 and shall use throughout this chapter. Consider the goal of estimating the distribution of radon levels of the houses within each of the 85 counties in Minnesota.[1] This seems

[1] Radon levels are always positive, and it is reasonable to suppose that effects will be multiplicative; hence it is appropriate to model the data on the logarithmic scale (see Section 4.4). For some purposes, though, such as estimating total cancer risk, it makes sense to estimate averages on the original, unlogged scale; we can obtain these inferences using simulation, as discussed at the end of Section 12.8.

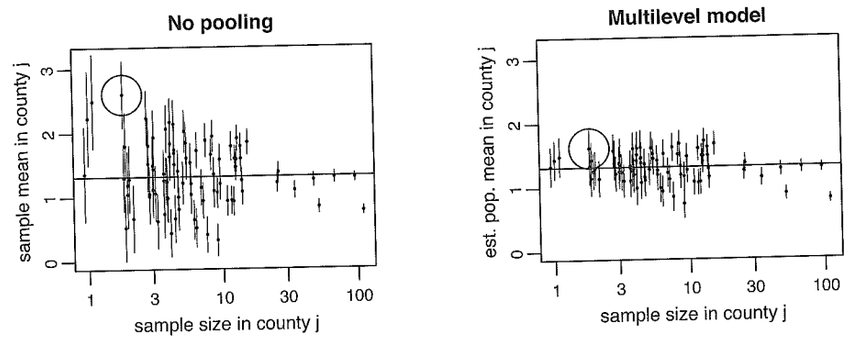

Figure 12.1 *Estimates ± standard errors for the average log radon levels in Minnesota counties plotted versus the (jittered) number of observations in the county: (a) no-pooling analysis, (b) multilevel (partial pooling) analysis, in both cases with no house-level or county-level predictors. The counties with fewer measurements have more variable estimates and larger higher standard errors. The horizontal line in each plot represents an estimate of the average radon level across all counties. The left plot illustrates a problem with the no-pooling analysis: it systematically causes us to think that certain counties are more extreme, just because they have smaller sample sizes.*

simple enough. One estimate would be the average that completely pools data across all counties. This ignores variation among counties in radon levels, however, so perhaps a better option would be simply to use the average log radon level in each county. Figure 12.1a plots these averages against the number of observations in each county.

Whereas complete pooling ignores variation between counties, the no-pooling analysis overstates it. To put it another way, the no-pooling analysis overfits the data within each county. To see this, consider Lac Qui Parle County (circled in the plot), which has the highest average radon level of all 85 counties in Minnesota. This average, however, is estimated using only two data points. Lac Qui Parle may very well be a high-radon county, but do we really believe it is *that* high? Maybe, but probably not: given the variability in the data we would not have much trust in an estimate based on only two measurements.

To put it another way, looking at all the counties together: the estimates from the no-pooling model overstate the variation among counties and tend to make the individual counties look more different than they actually are.

Partial-pooling estimates from a multilevel model

The multilevel estimates of these averages, displayed in Figure 12.1b, represent a compromise between these two extremes. The goal of estimation is the average log radon level α_j among all the houses in county j, for which all we have available are a random sample of size n_j. For this simple scenario with no predictors, the multilevel estimate for a given county j can be approximated as a weighted average of the mean of the observations in the county (the unpooled estimate, $\bar{y}_j$) and the mean over all counties (the completely pooled estimate, $\bar{y}_{\text{all}}$):

$$\hat{\alpha}_j^{\text{multilevel}} \approx \frac{\frac{n_j}{\sigma_y^2}\bar{y}_j + \frac{1}{\sigma_\alpha^2}\bar{y}_{\text{all}}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}, \tag{12.1}$$

where n_j is the number of measured houses in county j, σ_y^2 is the within-county variance in log radon measurements, and σ_α^2 is the variance among the average log radon levels of the different counties. We could also allow the within-county variance to vary by county (in which case σ_y would be replaced by $\sigma_{y\,j}$ in the preceding formula) but for simplicity we assume it is constant.

The weighted average (12.1) reflects the relative amount of information available about the individual county, on one hand, and the average of all the counties, on the other:

- Averages from counties with smaller sample sizes carry less information, and the weighting pulls the multilevel estimates closer to the overall state average. In the limit, if $n_j = 0$, the multilevel estimate is simply the overall average, $\bar{y}_{\text{all}}$.
- Averages from counties with larger sample sizes carry more information, and the corresponding multilevel estimates are close to the county averages. In the limit as $n_j \to \infty$, the multilevel estimate is simply the county average, $\bar{y}_j$.
- In intermediate cases, the multilevel estimate lies between the two extremes.

To actually apply (12.1), we need estimates of the variation within and between counties. In practice, we estimate these variance parameters together with the α_j's, either with an approximate program such as `lmer()` (see Section 12.4) or using fully Bayesian inference, as implemented in Bugs and described in Part 2B of this book. For now, we present inferences (as in Figure 12.1) without dwelling on the details of estimation.

12.3 Partial pooling with predictors

The same principle of finding a compromise between the extremes of complete pooling and no pooling applies for more general models. This section considers partial pooling for a model with unit-level predictors. In this scenario, no pooling might refer to fitting a separate regression model within each group. However, a less extreme and more common option that we also sometimes refer to as "no pooling" is a model that includes group indicators and estimates the model classically.[2]

As we move on to more complicated models, we present estimates graphically but do not continue with formulas of the form (12.1). However, the general principle remains that multilevel models compromise between pooled and unpooled estimates, with the relative weights determined by the sample size in the group and the variation within and between groups.

Complete-pooling and no-pooling analyses for the radon data, with predictors

Continuing with the radon data, Figure 12.2 shows the logarithm of the home radon measurement versus floor of measurement[3] for houses sampled from eight of the 85 counties in Minnesota. (We fit our model to the data from all 85 counties, including a total of 919 measurements, but to save space we display the data and estimates for a selection of eight counties, chosen to capture a range of the sample sizes in the survey.)

In each graph of Figure 12.2, the dashed line shows the linear regression of log

[2] This version of "no pooling" does not pool the estimates for the intercepts—the parameters we focus on in the current discussion—but it does completely pool estimates for any slope coefficients (they are forced to have the same value across all groups) and also assumes the residual variance is the same within each group.

[3] Measurements were taken in the lowest living area of each house, with basement coded as 0 and first floor coded as 1.

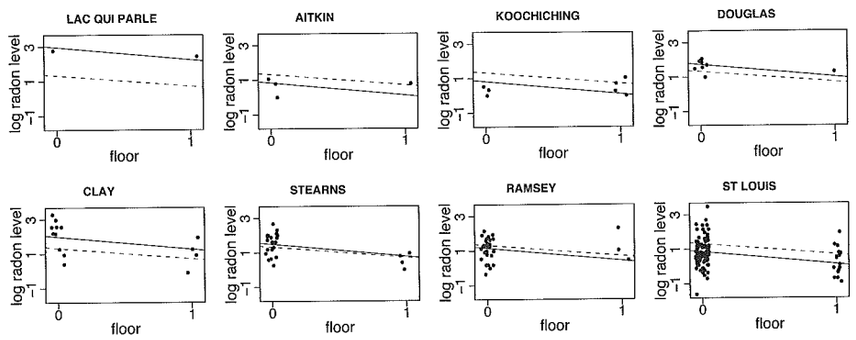

Figure 12.2 *Complete-pooling (dashed lines, $y = \alpha + \beta x$) and no-pooling (solid lines, $y = \alpha_j + \beta x$) regressions fit to radon data from the 85 counties in Minnesota, and displayed for eight of the counties. The estimated slopes β differ slightly for the two models, but here our focus is on the intercepts.*

radon, given the floor of measurement, using a model that pools all counties together (so the same line appears in all eight plots), and the solid line shows the no-pooling regressions, obtained by including county indicators in the regression (with the constant term removed to avoid collinearity; we also could have kept the constant term and included indicators for all but one of the counties). We can write the complete-pooling regression as $y_i = \alpha + \beta x_i + \epsilon_i$ and the no-pooling regression as $y_i = \alpha_{j[i]} + \beta x_i + \epsilon_i$, where $j[i]$ is the county corresponding to house i. The solid lines then plot $y = \hat{\alpha} + \hat{\beta} x$ from the complete-pooling model, and the dashed lines show $y = \hat{\alpha}_j + \hat{\beta} x$, for $j = 1, \ldots, 8$, from the no-pooling model.

Here is the complete-pooling regression for the radon data:

R output
```
lm(formula = y ~ x)
            coef.est coef.se
(Intercept)  1.33     0.03
x           -0.61     0.07
  n = 919, k = 2
  residual sd = 0.82
```

To fit the no-pooling model in R, we include the county index (a variable named `county` that takes on values between 1 and 85) as a factor in the regression—thus, predictors for the 85 different counties. We add "−1" to the regression formula to remove the constant term, so that all 85 counties are included. Otherwise, R would use county 1 as a baseline.

R output
```
lm(formula = y ~ x + factor(county) - 1)
                  coef.est coef.sd
x                 -0.72     0.07
factor(county)1    0.84     0.38
factor(county)2    0.87     0.10
. . .
factor(county)85   1.19     0.53
  n = 919, k = 86
  residual sd = 0.76
```

The estimated slopes β differ slightly for the two regressions. The no-pooling model includes county indicators, which can change the estimated coefficient for x, if the proportion of houses with basements varies among counties. This is just

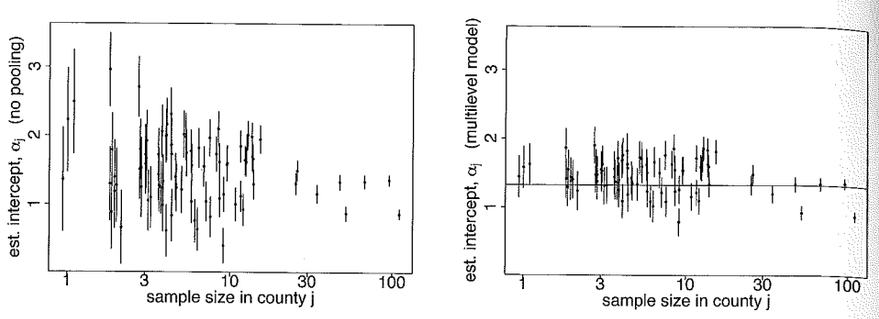

Figure 12.3 *(a) Estimates ± standard errors for the county intercepts α_j in the model $y_i = \alpha_{j[i]} + \beta x_i + error_i$, for the no-pooling analysis of the radon data, plotted versus number of observations from the county. The counties with fewer measurements have more variable estimates with higher standard errors. This graph illustrates a problem with classical regression: it systematically causes us to think that certain counties are more extreme, just because they have smaller sample sizes.*
(b) Multilevel (partial pooling) estimates ± standard errors for the county intercepts α_j for the radon data, plotted versus number of observations from the county. The horizontal line shows the complete pooling estimate. Comparing to the left plot (no pooling), which is on the same scale, we see that the multilevel estimate is typically closer to the complete-pooling estimate for counties with few observations, and closer to the no-pooling estimates for counties with many observations.
These plots differ only slightly from the no-pooling and multilevel estimates without the house-level predictor, as displayed in Figure 12.1.

a special case of the rule that adding new predictors in a regression can change the estimated coefficient of x, if these new predictors are correlated with x. In the particular example shown in Figure 12.2, the complete-pooling and no-pooling estimates of β differ only slightly; in the graphs, the difference can be seen most clearly in Stearns and Ramsey counties.

Problems with the no-pooling and complete-pooling analyses

Both the analyses shown in Figure 12.2 have problems. The complete-pooling analysis ignores any variation in average radon levels between counties. This is undesirable, particularly since the goal of our analysis was to identify counties with high-radon homes. We do not want to pool away the main subject of our study!

The no-pooling analysis has problems too, however, which we can again see in Lac Qui Parle County. Even after controlling for the floors of measurement, this county has the highest fitted line (that is, the highest estimate $\hat{\alpha}_j$), but again we do not have much trust in an estimate based on only two observations.

More generally, we would expect the counties with the least data to get more extreme estimates $\hat{\alpha}_j$ in the no-pooling analyses. Figure 12.3a illustrates with the estimates ± standard errors for the county intercepts α_j, plotted versus the sample size in each county j.

Multilevel analysis

The simplest multilevel model for the radon data with the floor predictor can be written as

$$y_i \sim \mathrm{N}(\alpha_{j[i]} + \beta x_i, \sigma_y^2), \quad \text{for } i = 1, \ldots, n, \tag{12.2}$$

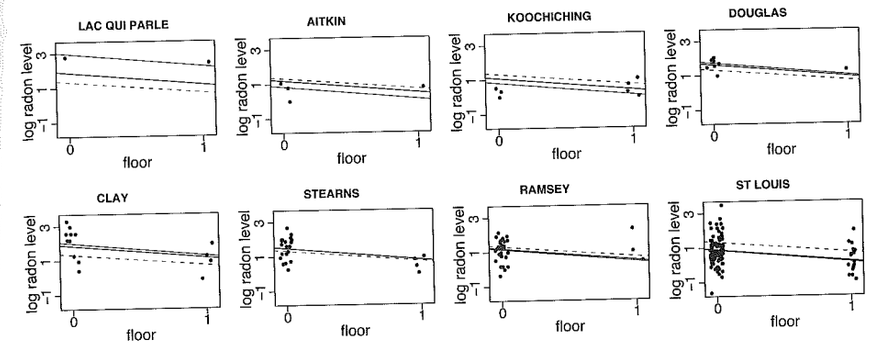

Figure 12.4 *Multilevel (partial pooling) regression lines $y = \alpha_j + \beta x$ fit to radon data from Minnesota, displayed for eight counties. Light-colored dashed and solid lines show the complete-pooling and no-pooling estimates, respectively, from Figure 12.3a.*

which looks like the no-pooling model but with one key difference. In the no-pooling model, the α_j's are set to the classical least squares estimates, which correspond to the fitted intercepts in a model run separately in each county (with the constraint that the slope coefficient equals β in all models). Model (12.2) also looks a little like the complete-pooling model except that, with complete pooling, the α_j's are given a "hard constraint"—they are all fixed at a common α.

In the multilevel model, a "soft constraint" is applied to the α_j's: they are assigned a probability distribution,

$$\alpha_j \sim \mathrm{N}(\mu_\alpha, \sigma_\alpha^2), \quad \text{for } j = 1, \ldots, J, \tag{12.3}$$

with their mean μ_α and standard deviation σ_α estimated from the data. The distribution (12.3) has the effect of pulling the estimates of α_j toward the mean level μ_α, but not all the way—thus, in each county, a *partial-pooling* compromise between the two estimates shown in Figure 12.2. In the limit of $\sigma_\alpha \to \infty$, the soft constraints do nothing, and there is no pooling; as $\sigma_\alpha \to 0$, they pull the estimates all the way to zero, yielding the complete-pooling estimate.

Figure 12.4 shows, for the radon example, the estimated line from the multilevel model (12.2), which in each county lies between the complete-pooling and no-pooling regression lines. There is strong pooling (solid line closer to complete-pooling line) in counties with small sample sizes, and only weak pooling (solid line closer to no-pooling line) in counties containing many measurements.

Going back to Figure 12.3, the right panel shows the estimates and standard errors for the county intercepts α_j from the multilevel model, plotted versus county sample size. Comparing to the left panel, we see more pooling for the counties with fewer observations. We also see a trend that counties with larger sample sizes have lower radon levels, indicating that "county sample size" is correlated with some relevant county-level predictor.

Average regression line and individual- and group-level variances

Multilevel models typically have so many parameters that it is not feasible to closely examine all their numerical estimates. Instead we plot the estimated group-level models (as in Figure 12.4) and varying parameters (as in Figure 12.3b) to look for patterns and facilitate comparisons across counties. It can be helpful, however,

to look at numerical summaries for the *hyperparameters*—those model parameters without group-level subscripts.

For example, in the radon model, the hyperparameters are estimated as $\hat{\mu}_\alpha = 1.46$, $\hat{\beta} = -0.69$, $\hat{\sigma}_y = 0.76$, and $\hat{\sigma}_\alpha = 0.33$. (We show the estimates in Section 12.4.) That is, the estimated average regression line for all the counties is $y = 1.46 - 0.69x$, with error standard deviations of 0.76 at the individual level and 0.33 at the county level. For this dataset, variation within counties (after controlling for the floor of measurement) is comparable to the average difference between measurements in houses with and without basements.

One way to interpret the variation between counties, σ_α, is to consider the variance ratio, $\sigma_\alpha^2/\sigma_y^2$, which in this example is estimated at $0.33^2/0.76^2 = 0.19$, or about one-fifth. Thus, the standard deviation of average radon levels between counties is the same as the standard deviation of the average of 5 measurements within a county (that is, $0.76/\sqrt{5} = 0.33$). The relative values of individual- and group-level variances are also sometimes expressed using the *intraclass correlation*, $\sigma_\alpha^2/(\sigma_\alpha^2 + \sigma_y^2)$, which ranges from 0 if the grouping conveys no information to 1 if all members of a group are identical.

In our example, the group-level model tells us that the county intercepts, α_j, have an estimated mean of 1.46 and standard deviation of 0.33. (What is relevant to our discussion here is the standard deviation, not the mean.) The amount of information in this distribution is the same as that in 5 measurements within a county. To put it another way, for a county with a sample size less than 5, there is more information in the group-level model than in the county's data; for a county with more than 5 observations, the within-county measurements are more informative (in the sense of providing a lower-variance estimate of the county's average radon level). As a result, the multilevel regression line in a county is closer to the complete-pooling estimate when sample size is less than 5, and closer to the no-pooling estimate when sample size exceeds 5. We can see this in Figure 12.4: as sample size increases, the multilevel estimates move closer and closer to the no-pooling lines.

Partial pooling (shrinkage) of group coefficients α_j

Multilevel modeling partially pools the group-level parameters α_j toward their mean level, μ_α. There is more pooling when the group-level standard deviation σ_α is small, and more smoothing for groups with fewer observations. Generalizing (12.1), the multilevel-modeling estimate of α_j can be expressed as a weighted average of the no-pooling estimate for its group ($\bar{y}_j - \beta\bar{x}_j$) and the mean, μ_α:

$$\text{estimate of } \alpha_j \approx \frac{\frac{n_j}{\sigma_y^2}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}(\bar{y}_j - \beta\bar{x}_j) + \frac{\frac{1}{\sigma_\alpha^2}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}\mu_\alpha. \qquad (12.4)$$

When actually fitting multilevel models, we do not actually use this formula; rather, we fit models using `lmer()` or Bugs, which automatically perform the calculations, using formulas such as (12.4) internally. Chapter 19 provides more detail on the algorithms used to fit these models.

Classical regression as a special case

Classical regression models can be viewed as special cases of multilevel models. The limit of $\sigma_\alpha \to 0$ yields the complete-pooling model, and $\sigma_\alpha \to \infty$ reduces to the no-pooling model. Given multilevel data, we can estimate σ_α. Therefore we

see no reason (except for convenience) to accept estimates that arbitrarily set this parameter to one of these two extreme values.

12.4 Quickly fitting multilevel models in R

We fit most of the multilevel models in this part of the book using the `lmer()` function, which fits linear and generalized linear models with varying coefficients.[4] Part 2B of the book considers computation in more detail, including a discussion of why it can be helpful to make the extra effort and program models using Bugs (typically using a simpler `lmer()` fit as a starting point). The `lmer()` function is currently part of the R package `Matrix`; see Appendix C for details. Here we introduce `lmer()` in the context of simple varying-intercept models.

The lmer function

Varying-intercept model with no predictors. The varying intercept model with no predictors (discussed in Section 12.2) can be fit and displayed using `lmer()` as follows:

R code
```
M0 <- lmer (y ~ 1 + (1 | county))
display (M0)
```

This model simply includes a constant term (the predictor "1") and allows it to vary by county. We next move to a more interesting model including the floor of measurement as an individual-level predictor.

Varying-intercept model with an individual-level predictor. We shall introduce multilevel fitting with model (12.2)–(12.3), the varying-intercept regression with a single predictor. We start with the call to `lmer()`:

R code
```
M1 <- lmer (y ~ x + (1 | county))
```

This expression starts with the no-pooling model, "`y ~ x`," and then adds "`(1 | county)`," which allows the intercept (the coefficient of the predictor "1," which is the column of ones—the constant term in the regression) to vary by county.

We can then display a quick summary of the fit:

R code
```
display (M1)
```

which yields

R output
```
lmer(formula = y ~ x + (1 | county))
            coef.est coef.se
(Intercept)  1.46     0.05
x           -0.69     0.07
Error terms:
 Groups   Name        Std.Dev.
 county   (Intercept) 0.33
 Residual             0.76
# of obs: 919, groups: county, 85
deviance = 2163.7
```

[4] The name `lmer` stands for "linear mixed effects in R," but the function actually works for generalized linear models as well. The term "mixed effects" refers to random effects (coefficients that vary by group) and fixed effects (coefficients that do not vary). We avoid the terms "fixed" and "random" (see page 245) and instead refer to coefficients as "modeled" (that is, grouped) or "unmodeled."

The top part of this display shows the inference about the intercept and slope for the model, averaging over the counties. The bottom part gives the estimated variation: $\hat{\sigma}_\alpha = 0.33$ and $\hat{\sigma}_y = 0.76$. We also see that the model was fit to 919 houses within 85 counties. We shall ignore the deviance for now.

Estimated regression coefficients

To see the estimated model within each county. We type

R code
```
coef (M1)
```

which yields

R output
```
$county
   (Intercept)     x
1         1.19 -0.69
2         0.93 -0.69
3         1.48 -0.69
. . .
85        1.39 -0.69
```

Thus, the estimated regression line is $y = 1.19 - 0.69x$ in county 1, $y = 0.93 + 0.69x$ in county 2, and so forth. The slopes are all identical because they were specified thus in the model. (The specification `(1|county)` tells the model to allow only the intercept to vary. As we shall discuss in the next chapter, we can allow the slope to vary by specifying `(1+x|county)` in the regression model.)

Fixed and random effects. Alternatively, we can separately look at the estimated model averaging over the counties—the "fixed effects"—and the county-level errors—the "random effects." Typing

R code
```
fixef (M1)
```

yields

R output
```
(Intercept)           x
       1.46       -0.69
```

The estimated regression line in an average county is thus $y = 1.46 - 0.69x$. We can then look at the county-level errors:

R code
```
ranef (M1)
```

which yields

R output
```
   (Intercept)
1        -0.27
2        -0.53
3         0.02
. . .
85       -0.08
```

These tell us how much the intercept is shifted up or down in particular counties. Thus, for example, in county 1, the estimated intercept is 0.27 lower than average, so that the regression line is $(1.46 - 0.27) - 0.69x = 1.19 - 0.69x$, which is what we saw earlier from the call to `coef()`. For some applications, it is best to see the estimated model within each group; for others, it is helpful to see the estimated average model and group-level errors.

Uncertainties in the estimated coefficients

We wrote little functions `se.fixef()` and `se.ranef()` for quickly pulling out these standard errors from the model fitted by `lmer()`. In this example,

R code
```
se.fixef (M1)
```

yields

R output
```
(Intercept)           x
       0.05        0.07
```

and

R code
```
se.ranef (M1)
```

yields,

R output
```
$county
   (Intercept)
1         0.25
2         0.10
3         0.26
. . .
85        0.28
```

As discussed in Section 12.3, the standard errors differ according to the sample size within each county; for example, counties 1, 2, and 85 have 4, 52, and 2 houses, respectively, in the sample. For the within-county regressions, standard errors are only given for the intercepts, since this model has a common slope for all counties.

Summarizing and displaying the fitted model

We can access the components of the estimates and standard errors using list notation in R. For example, to get a 95% confidence interval for the slope (which, in this model, does not vary by county):

R code
```
fixef(M1)["x"] + c(-2,2)*se.fixef(M1)["x"]
```

or, equivalently, since the slope is the second coefficient in the regression,

R code
```
fixef(M1)[2] + c(-2,2)*se.fixef(M1)[2]
```

The term "fixed effects" is used for the regression coefficients that do not vary by group (such as the coefficient for x in this example) or for group-level coefficients or group averages (such as the average intercept, μ_α in (12.3)).

Identifying the batches of coefficients. In pulling out elements of the coefficients from `coef()` or `ranef()`, we must first identify the grouping (`county`, in this case). The need for this labeling will become clear in the next chapter in the context of non-nested models, where there are different levels of grouping and thus different structures of varying coefficients.

For example, here is a 95% confidence interval for the intercept in county 26:

R code
```
coef(M1)$county[26,1] + c(-2,2)*se.ranef(M1)$county[26]
```

and here is a 95% confidence interval for the error in the intercept in that county (that is, the deviation from the average):

R code
```
as.matrix(ranef(M1)$county)[26] + c(-2,2)*se.ranef(M1)$county[26]
```

For a more elaborate example, we make Figure 12.4 using the following commands:

```
                coef(M1)$county[,1]    # 1st column is the intercept
                oef(M1)$county[,2]     # 2nd element is the slope
                + runif(n,-.05,.05)      # jittered data for plotting
                  ))                     # make a 2x4 grid of plots
                 ay8){
              tter[county==j], y[county==j], xlim=c(-.05,1.05),
            =y.range, xlab="floor", ylab="log radon level", main=uniq.name[j])
          uniq.name is a vector of county names that was created earlier]
        curve (coef(lm.pooled)[1] + coef(lm.pooled)[2]*x, lty=2, col="gray10",
          add=TRUE)
        curve (coef(lm.unpooled)[j+1] + coef(lm.unpooled)[1]*x, col="gray10",
          add=TRUE)
        curve (a.hat.M1[j] + b.hat.M1[j]*x, lwd=1, col="black", add=TRUE)
    }
```

Here, `lm.pooled` and `lm.unpooled` are the classical regressions that we have already fit.

More complicated models

The `lmer()` function can also handle many of the multilevel regressions discussed in this part of the book, including group-level predictors, varying intercepts and slopes, nested and non-nested structures, and multilevel generalized linear models. Approximate routines such as `lmer()` tend to work well when the sample size and number of groups is moderate to large, as in the radon models. When the number of groups is small, or the model becomes more complicated, it can be useful to switch to Bayesian inference, using the Bugs program, to better account for uncertainty in model fitting. We return to this point in Section 16.1.

12.5 Five ways to write the same model

We begin our treatment of multilevel models with the simplest structures—*nested* models, in which we have observations $i = 1, \ldots, n$ clustered in groups $j = 1, \ldots, J$, and we wish to model variation among groups. Often, predictors are available at the individual and group levels. We shall use as a running example the home radon analysis described above, using as predictors the house-level x_i and a measure of the logarithm of soil uranium as a county-level predictor, u_j. For some versions of the model, we include these both as individual-level predictors and label them as X_{i1} and X_{i2}.

There are several different ways of writing a multilevel model. Rather than introducing a restrictive uniform notation, we describe these different formulations and explain how they are connected. It is useful to be able to express a model in different ways, partly so that we can recognize the similarities between models that only appear to be different, and partly for computational reasons.

Allowing regression coefficients to vary across groups

Perhaps the simplest way to express a multilevel model generally is by starting with the classical regression model fit to all the data, $y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \epsilon_i$, and then generalizing to allow the coefficients β to vary across groups; thus,

$$y_i = \beta_{0\,j[i]} + \beta_{1\,j[i]} X_{i1} + \beta_{2\,j[i]} X_{i2} + \cdots + \epsilon_i.$$

The "multilevel" part of the model involves assigning a multivariate distribution to the vector of β's within each group, as we discuss in Section 13.1.

For now we will focus on *varying-intercept models*, in which the only coefficient that varies across groups is the constant term β_0 (which, to minimize subscripting, we label α). For the radon data that include the floor and a county-level uranium predictor, the model then becomes

$$y_i = \alpha_{j[i]} + \beta_1 X_{i1} + \beta_2 X_{i2} + \epsilon_i$$

where X_{i1} is the i^{th} element of the vector $X_{(1)}$ representing the first-floor indicators and X_{i2} is the i^{th} element of the vector $X_{(2)}$ representing the uranium measurement in the county containing house i. We can also write this in matrix notation as

$$y_i = \alpha_{j[i]} + X_i \beta + \epsilon_i$$

with the understanding that X includes the first-floor indicator and the county uranium measurement but not the constant term. This is the way that models are built using `lmer()`, including all predictors at the individual level, as we discuss in Section 12.6.

The second level of the model is simply

$$\alpha_j \sim \text{N}(\mu_\alpha, \sigma_\alpha^2). \tag{12.5}$$

Group-level errors. The model (12.5) can also be written as

$$\alpha_j = \mu_\alpha + \eta_j, \quad \text{with } \eta_j \sim \text{N}(0, \sigma_\alpha^2). \tag{12.6}$$

The group-level errors η_j can be helpful in understanding the model; however, we often use the more compact notation (12.5) to reduce the profusion of notation. (We have also toyed with notation such as $\alpha_j = \mu^\alpha + \epsilon_j^\alpha$ in which ϵ is consistently used for regression errors—but the superscripts seem too confusing. As illustrated in Part 2B of this book, we sometimes use such notation when programming models in Bugs.)

Combining separate local regressions

An alternative way to write the multilevel model is as a linking of local regressions in each group. Within each group j, a regression is performed on the local predictors (in this case, simply the first-floor indicator, x_i), with a constant term α that is indexed by group:

$$\text{within county } j\text{:} \quad y_i \sim \text{N}(\alpha_j + \beta x_i, \sigma_y^2), \quad \text{for } i = 1, \ldots, n_j. \tag{12.7}$$

The county uranium measurement has not yet entered the model since we are imagining separate regressions fit to each county—there would be no way to estimate the coefficient for a county-level predictor from any of these within-county regressions.

Instead, the county-level uranium level, u_j, is included as a predictor in the second level of the model:

$$\alpha_j \sim \text{N}(\gamma_0 + \gamma_1 u_j, \sigma_\alpha^2). \tag{12.8}$$

We can also write the distribution in (12.8) as $\text{N}(U_j \gamma, \sigma_\alpha^2)$, where U has two columns: a constant term, $U_{(0)}$, and the county-level uranium measurement, $U_{(1)}$. The errors in this model (with mean 0 and standard deviation σ_α) represent variation *among counties* that is not explained by the local and county-level predictors.

The multilevel model combines the J local regression models (12.7) in two ways: first, the local regression coefficients β are the same in all J models (an assumption we will relax in Section 13.1). Second, the different intercepts α_j are connected through the group-level model (12.8), with consequences to the coefficient estimates that we discuss in Section 12.6.

Group-level errors. We can write (12.8) as

$$\alpha_j = \gamma_0 + \gamma_1 u_j + \eta_j, \quad \text{with } \eta_j \sim \text{N}(0, \sigma_\alpha^2), \tag{12.9}$$

explicitly showing the errors in the county-level regression.

Modeling the coefficients of a large regression model

The identical model can be written as a single regression, in which the local and group-level predictors are combined into a single matrix X:

$$y_i \sim \text{N}(X_i\beta, \sigma_y^2), \tag{12.10}$$

where, for our example, X includes vectors corresponding to:

- A constant term, $X_{(0)}$;
- The floor where the measurement was taken, $X_{(1)}$;
- The county-level uranium measure, $X_{(2)}$;
- J (not $J-1$) county indicators, $X_{(3)}, \ldots, X_{(J+2)}$.

At the upper level of the model, the J county indicators (which in this case are $\beta_3, \ldots, \beta_{J+2}$) follow a normal distribution:

$$\beta_j \sim \text{N}(0, \sigma_\alpha^2), \quad \text{for } j = 3, \ldots, J+2. \tag{12.11}$$

In this case, we have centered the β_j distribution at 0 rather than at an estimated μ_β because any such μ_β would be statistically indistinguishable from the constant term in the regression. We return to this point shortly.

The parameters in the model (12.10)–(12.11) can be identified exactly with those in the separate local regressions above:

- The local predictor x in model (12.7) is the same as $X_{(1)}$ (the floor) here.
- The local errors ϵ_i are the same in the two models.
- The matrix of group-level predictors U in (12.8) is just $X_{(0)}$ here (the constant term) joined with $X_{(2)}$ (the uranium measure).
- The group-level errors $\eta_1, \ldots, \eta_J$ in (12.9) are identical to $\beta_3, \ldots, \beta_{J+2}$ here.
- The standard-deviation parameters σ_y and σ_α keep the same meanings in the two models.

Moving the constant term around. The multilevel model can be written in yet another equivalent way by moving the constant term:

$$\begin{aligned} y_i &= \text{N}(X_i\beta, \sigma_y^2), \quad \text{for } i = 1, \ldots, n \\ \beta_j &\sim \text{N}(\mu_\alpha, \sigma_\alpha^2), \quad \text{for } j = 3, \ldots, J+2. \end{aligned} \tag{12.12}$$

In this version, we have removed the constant term from X (so that it now has only $J+2$ columns) and replaced it by the equivalent term μ_α in the group-level model. The coefficients $\beta_3, \ldots, \beta_{J+2}$ for the group indicators are now centered around μ_α rather than 0, and are equivalent to $\alpha_1, \ldots, \alpha_J$ as defined earlier, for example, in model (12.9).

Regression with multiple error terms

Another option is to re-express model (12.10), treating the group-indicator coefficients as error terms rather than regression coefficients, in what is often called a "mixed effects" model popular in the social sciences:

$$\begin{aligned} y_i &\sim \text{N}(X_i\beta + \eta_{j[i]}, \sigma_y^2), \quad \text{for } i = 1, \ldots, n \\ \eta_j &\sim \text{N}(0, \sigma_\alpha^2), \end{aligned} \tag{12.13}$$

where $j[i]$ represents the county that contains house i, and X now contains only three columns:

- A constant term, $X_{(0)}$;
- The floor, $X_{(1)}$;
- The county-level uranium measure, $X_{(2)}$.

This is the same as model (12.10)–(12.11), simply renaming some of the β_j's as η_j's. All our tools for multilevel modeling will automatically work for models with multiple error terms.

Large regression with correlated errors

Finally, we can express a multilevel model as a classical regression with correlated errors:

$$y_i = X_i\beta + \epsilon_i^{\text{all}}, \quad \epsilon^{\text{all}} \sim \text{N}(0, \Sigma), \tag{12.14}$$

where X is now the matrix with three predictors (the constant term, first-floor indicator, and county-level uranium measure) as in (12.13), but now the errors ϵ_i^{all} have an $n \times n$ covariance matrix Σ. The error ϵ_i^{all} in (12.14) is equivalent to the sum of the two errors, $\eta_{j[i]} + \epsilon_i$, in (12.13). The term $\eta_{j[i]}$, which is the same for all units i in group j, induces correlation in ϵ^{all}.

In multilevel models, Σ is parameterized in some way, and these parameters are estimated from the data. For the nested multilevel model we have been considering here, the variances and covariances of the n elements of ϵ^{all} can be derived in terms of the parameters σ_y and σ_α:

$$\begin{aligned} \text{For any unit } i: \quad & \Sigma_{ii} = \text{var}(\epsilon_i^{\text{all}}) = \sigma_y^2 + \sigma_\alpha^2 \\ \text{For any units } i, k \text{ within the same group } j: \quad & \Sigma_{ik} = \text{cov}(\epsilon_i^{\text{all}}, \epsilon_k^{\text{all}}) = \sigma_\alpha^2 \\ \text{For any units } i, k \text{ in different groups:} \quad & \Sigma_{ik} = \text{cov}(\epsilon_i^{\text{all}}, \epsilon_k^{\text{all}}) = 0. \end{aligned}$$

It can also be helpful to express Σ in terms of standard errors and correlations:

$$\begin{aligned} \text{sd}(\epsilon_i) &= \sqrt{\Sigma_{ii}} = \sqrt{\sigma_y^2 + \sigma_\alpha^2} \\ \text{corr}(\epsilon_i, \epsilon_k) &= \frac{\Sigma_{ik}}{\sqrt{\Sigma_{ii}\Sigma_{kk}}} = \begin{cases} \frac{\sigma_\alpha^2}{\sigma_y^2 + \sigma_\alpha^2} & \text{if } j[i] = j[k] \\ 0 & \text{if } j[i] \neq j[k]. \end{cases} \end{aligned}$$

We generally prefer modeling the multilevel effects explicitly rather than burying them as correlations, but once again it is useful to see how the same model can be written in different ways.

12.6 Group-level predictors

Adding a group-level predictor to improve inference for group coefficients α_j

We continue with the radon example from Sections 12.2–12.3 to illustrate how a multilevel model handles predictors at the group as well as the individual levels.

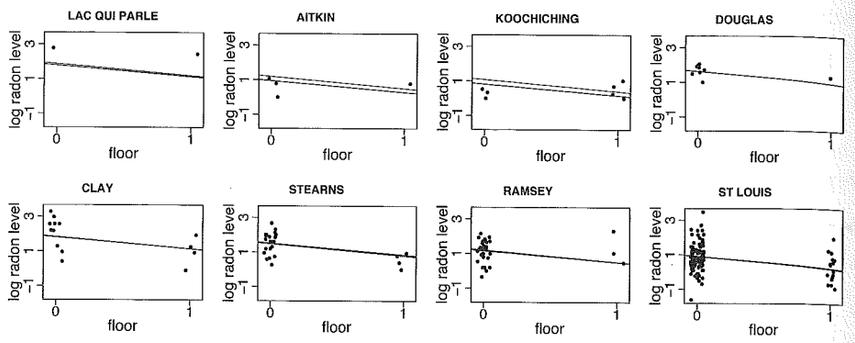

Figure 12.5 *Multilevel (partial pooling) regression lines $y = \alpha_j + \beta x$ fit to radon data, displayed for eight counties, including uranium as a county-level predictor. Light-colored lines show the multilevel estimates, without uranium as a predictor, from Figure 12.4.*

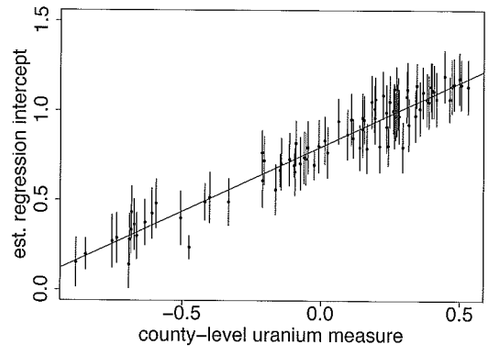

Figure 12.6 *Estimated county coefficients α_j (± 1 standard error) plotted versus county-level uranium measurement u_j, along with the estimated multilevel regression line $\alpha_j = \gamma_0 + \gamma_1 u_j$. The county coefficients roughly follow the line but not exactly; the deviation of the coefficients from the line is captured in σ_α, the standard deviation of the errors in the county-level regression.*

We use the formulation

$$\begin{aligned} y_i &\sim \text{N}(\alpha_{j[i]} + \beta x_i,\, \sigma_y^2), \text{ for } i = 1, \ldots, n \\ \alpha_j &\sim \text{N}(\gamma_0 + \gamma_1 u_j,\, \sigma_\alpha^2), \text{ for } j = 1, \ldots, J, \end{aligned} \tag{12.15}$$

where x_i is the house-level first-floor indicator and u_j is the county-level uranium measure.

R code

```
u.full <- u[county]
M2 <- lmer (y ~ x + u.full + (1 | county))
display (M2)
```

This model includes floor, uranium, and intercepts that vary by county. The `lmer()` function only accepts predictors at the individual level, so we have converted u_j to $u_i^{\text{full}} = u_{j[i]}$ (with the variable `county` playing the role of the indexing $j[i]$), to pull out the uranium level of the county where house i is located.

The display of the `lmer()` fit shows coefficients and standard errors, along with estimated residual variation at the county and individual ("residual") level:

R output

```
lmer(formula = y ~ x + u.full + (1 | county))
            coef.est coef.se
(Intercept)  1.47     0.04
x           -0.67     0.07
u.full       0.72     0.09
Error terms:
 Groups   Name        Std.Dev.
 county   (Intercept) 0.16
 Residual             0.76
# of obs: 919, groups: county, 85
deviance = 2122.9
```

As in our earlier example on page 261, we use `coef()` to pull out the estimated coefficients,

R code

```
coef (M2)
```

yielding

R output

```
$county
   (Intercept)     x u.full
1         1.45 -0.67   0.72
2         1.48 -0.67   0.72
. . .
85        1.42 -0.67   0.72
```

Only the intercept varies, so the coefficients for `x` and `u.full` are the same for all 85 counties. (Actually, `u.full` is constant within counties so it cannot have a varying coefficient here.) On page 280 we shall see a similar display for a model in which the coefficient for x varies by county.

As before, we can also examine the estimated model averaging over the counties:

R code

```
fixef (M2)
```

yielding

R output

```
(Intercept)           x      u.full
       1.47       -0.67        0.72
```

and the county-level errors:

R code

```
ranef (M2)
```

yielding

R output

```
   (Intercept)
1        -0.02
2         0.01
. . .
85       -0.04
```

The results of `fixef()` and `ranef()` add up to the coefficients in `coef()`: for county 1, $1.47 - 0.02 = 1.45$, for county 2, $1.47 + 0.01 = 1.48$, ..., and for county 85, $1.47 - 0.04 = 1.42$ (up to rounding error).

Interpreting the coefficients within counties

We can add the unmodeled coefficients (the "fixed effects") to the county-level errors to get an intercept and slope for each county. We start with the model that averages over all counties, $y_i = 1.47 - 0.67x_i + 0.72u_{j[i]}$ (as obtained from `display(M2)` or `fixef(M2)`.

Now consider a particular county, for example county 85. We can determine its fitted regression line in two ways from the `lmer()` output, in each case using the log uranium level in county 85, $u_{85} = 0.36$.

First, using the the last line of the display of `coef(M2)`, the fitted model for county 85 is $y_i = 1.42 - 0.67x_i + 0.72u_{85} = (1.42 + 0.72 \cdot 0.36) - 0.67x_i = 1.68 - 0.67x_i$, that is, 1.68 for a house with a basement and 1.01 for a house with no basement. Exponentiating gives estimated geometric mean predictions of 5.4 pCi/L and 2.7 pCi/L for houses in county 85 with and without basements.

Alternatively, we can construct the fitted line for county 85 by starting with the results from `fixef(M2)`—that is, $y_i = 1.47 - 0.67x_i + 0.72u_{j[i]}$, setting $u_{j[i]} = u_{85} = 0.36$—and adding the group-level error from `ranef(M2)`, which for county 85 is -0.04. The resulting model is $y_i = 1.47 - 0.67x_i + 0.72 \cdot 0.36 - 0.04 = 1.68 - 0.67x_i$, the same as in the other calculation (up to rounding error in the last digit of the intercept).

Figure 12.5 shows the fitted line for each of a selection of counties, and Figure 12.6 shows the county-level regression, plotting the estimated coefficients α_j versus the county-level predictor u_j. These two figures represent the two levels of the multilevel model.

The group-level predictor has increased the precision of our estimates of the county intercepts α_j: the ± 1 standard-error bounds are narrower in Figure 12.6 than in Figure 12.3b, which showed α_j's estimated without the uranium predictor (note the different scales on the y-axes of the two plots and the different county variables plotted on the x-axes).

The estimated individual- and county-level standard deviations in this model are $\hat{\sigma}_y = 0.76$ and $\hat{\sigma}_\alpha = 0.16$. In comparison, these residual standard deviations were 0.76 and 0.33 without the uranium predictor. This predictor has left the within-county variation unchanged—which makes sense, since it is a county-level predictor which has no hope of explaining variation within any county—but has drastically reduced the unexplained variation between counties. In fact, the variance ratio is now only $\sigma_\alpha^2/\sigma_y^2 = 0.16^2/0.76^2 = 0.044$, so that the county-level model is as good as $1/0.044 = 23$ observations within any county. The multilevel estimates under this new model will be close to the complete-pooling estimates (with county-level uranium included as a predictor) for many of the smaller counties in the dataset because a county would have to have more than 23 observations to be pulled closer to the no-pooling estimate than the complete-pooling estimate.

Interpreting the coefficient of the group-level predictor

The line in Figure 12.6 shows the prediction of average log radon in a county (for homes with basements—that is, $x_i = 0$—since these are the intercepts α_j), as a function of the log uranium level in the county. This estimated group-level regression line has an estimated slope of about 0.7. Coefficients between 0 and 1 are typical in a log-log regression: in this case, each increase of 1% in uranium level corresponds to a 0.7% predicted increase in radon.

It makes sense that counties higher in uranium have higher radon levels, and it also makes sense that the slope is less than 1. Radon is affected by factors other

than soil uranium, and the "uranium" variable in the dataset is itself an imprecise measure of actual soil uranium in the county, and so we would expect a 1% increase in the uranium variable to match to something less than a 1% increase in radon. Compared to classical regression, the estimation of this coefficient is trickier (since the α_j's—the "data" for the county-level regression—are not themselves observed) but the principles of interpretation do not change.

A multilevel model can include county indicators along with a county-level predictor

Users of multilevel models are often confused by the idea of including county indicators along with a county-level predictor. Is this possible? With 85 counties in the dataset, how can a regression fit 85 coefficients for counties, plus a coefficient for county-level uranium? This would seem to induce perfect collinearity into the regression or, to put it more bluntly, to attempt to learn more than the data can tell us. Is it really possible to estimate 86 coefficients from 85 data points?

The short answer is that we really have more than 85 data points. There are hundreds of houses with which to estimate the 85 county-level intercepts, and 85 counties with which to estimate the coefficient of county-level uranium. In a classical regression, however, the 85 county indicators and the county-level predictor would indeed be collinear. This problem is avoided in a multilevel model because of the partial pooling of the α_j's toward the group-level linear model. This is illustrated in Figure 12.6, which shows the estimates of all these 86 parameters—the 85 separate points and the slope of the line. In this model that includes a group-level predictor, the estimated intercepts are pulled toward this group-level regression line (rather than toward a constant, as in Figure 12.3b). The county-level uranium predictor u_j thus helps us estimate the county intercepts α_j but without overwhelming the information in individual counties.

Partial pooling of group coefficients α_j in the presence of group-level predictors

Equation (12.4) on page 258 gives the formula for partial pooling in the simple model with no group-level predictors. Once we add a group-level regression, $\alpha_j \sim \mathrm{N}(U_j\gamma, \sigma_\alpha^2)$, the parameters α_j are shrunk toward their regression estimates $\hat{\alpha}_j = U_j\gamma$. Equivalently, we can say that the group-level errors η_j (in the model $\alpha_j = U_j\gamma + \eta_j$) are shrunk toward 0. As always, there is more pooling when the group-level standard deviation σ_α is small, and more smoothing for groups with fewer observations. The multilevel estimate of α_j is a weighted average of the no-pooling estimate for its group ($\bar{y}_j - \overline{X}_j\beta$) and the regression prediction $\hat{\alpha}_j$:

$$\text{estimate of } \alpha_j \approx \frac{\frac{n_j}{\sigma_y^2}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}} \cdot (\text{estimate from group } j) + \frac{\frac{1}{\sigma_\alpha^2}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}} \cdot (\text{estimate from regression}). \quad (12.16)$$

Equivalently, the group-level errors η_j are partially pooled toward zero:

$$\text{estimate of } \eta_j \approx \frac{\frac{n_j}{\sigma_y^2}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}(\bar{y}_j - \overline{X}_j\beta - U_j\gamma) + \frac{\frac{1}{\sigma_\alpha^2}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}} \cdot 0.$$

12.7 Model building and statistical significance

From classical to multilevel regression

When confronted with a multilevel data structure, such as the radon measurements considered here or the examples in the previous chapter, we typically start by fitting some simple classical regressions and then work our way up to a full multilevel model. The four natural starting points are:

- Complete-pooling model: a single classical regression completely ignoring the group information—that is, a single model fit to all the data, perhaps including group-level predictors but with no coefficients for group indicators.
- No-pooling model: a single classical regression that includes group indicators (but no group-level predictors) but with no model for the group coefficients.
- Separate models: a separate classical regression in each group. This approach is not always possible if there are groups with small sample sizes. (For example, in Figure 12.4 on page 257, Aitkin County has three measurements in homes with basements and one in a home with no basement. If the sample from Aitkin County had happened to contain only houses with basements, then it would be impossible to estimate the slope β from this county alone.)
- Two-step analysis: starting with either the no-pooling or separate models, then fitting a classical group-level regression using, as "data," the estimated coefficients for each group.

Each of these simpler models can be informative in its own right, and they also set us up for understanding the partial pooling in a multilevel model, as in Figure 12.4.

For large datasets, fitting a model separately in each group can be computationally efficient as well. One might imagine an iterative procedure that starts by fitting separate models, continues with the two-step analysis, and then returns to fitting separate models, but using the resulting group-level regression to guide the estimates of the varying coefficients. Such a procedure, if formalized appropriately, is in fact the usual algorithm used to fit multilevel models, as we discuss in Chapter 17.

When is multilevel modeling most effective?

Multilevel model is most important when it is close to complete pooling, at least for some of the groups (as for Lac Qui Parle County in Figure 12.4 on page 257). In this setting we can allow estimates to vary by group while still estimating them precisely. As can be seen from formula (12.16), estimates are more pooled when the group-level standard deviation σ_α is small, that is, when the groups are similar to each other. In contrast, when σ_α is large, so that groups vary greatly, multilevel modeling is not much better than simple no-pooling estimation.

At this point, it might seem that we are contradicting ourselves. Earlier we motivated multilevel modeling as a compromise between no pooling and complete pooling, but now we are saying that multilevel modeling is effective when it is close to complete pooling, and ineffective when it is close to no pooling. If this is so, why not just always use the complete-pooling estimate?

We answer this question in two ways. First, when the multilevel estimate is close to complete pooling, it still allows variation between groups, which can be important, in fact can be one of the goals of the study. Second, as in the radon example, the multilevel estimate can be close to complete pooling for groups with small sample size and close to no pooling for groups with large sample size, automatically performing well for both sorts of group.

Using group-level predictors to make partial pooling more effective

In addition to being themselves of interest, group-level predictors play a special role in multilevel modeling by reducing the unexplained group-level variation and thus reducing the group-level standard deviation σ_α. This in turn increases the amount of pooling done by the multilevel estimate (see formula (12.16)), giving more precise estimates of the α_j's, especially for groups for which the sample size n_j is small. Following the template of classical regression, multilevel modeling typically proceeds by adding predictors at the individual and group levels and reducing the unexplained variance at each level. (However, as discussed in Section 21.7, adding a group-level predictor can actually increase the unexplained variance in some situations.)

Statistical significance

It is *not* appropriate to use statistical significance as a criterion for including particular group indicators in a multilevel model. For example, consider the simple varying-intercept radon model with no group-level predictor, in which the average intercept μ_α is estimated at 1.46, and the within-group intercepts α_j are estimated at $1.46 - 0.27 \pm 0.25$ for county 1, $1.46 - 0.53 \pm 0.10$ for county 2, $1.46 + 0.02 \pm 0.28$ for county 3, and so forth (see page 261).

County 1 is thus approximately 1 standard error away from the average intercept of 1.46, county 2 is more than 4 standard errors away, ... and county 85 is less than 1 standard error away. Of these three counties, only county 2 would be considered "statistically significantly" different from the average.

However, we should include all 85 counties in the model, and nothing is lost by doing so. The purpose of the multilevel model is not to see whether the radon levels in county 1 are statistically significantly different from those in county 2, or from the Minnesota average. Rather, we seek the best possible estimate in each county, with appropriate accounting for uncertainty. Rather than make some significance threshold, we allow all the intercepts to vary and recognize that we may not have much precision in many of the individual groups. We illustrate this point in another example in Section 21.8.

The same principle holds for the models discussed in the following chapters, which include varying slopes, non-nested levels, discrete data, and other complexities. Once we have included a source of variation, we do not use statistical significance to pick and choose indicators to include or exclude from the model.

In practice, our biggest constraints—the main reasons we do not use extremely elaborate models in which all coefficients can vary with respect to all grouping factors—are fitting and understanding complex models. The `lmer()` function works well when it works, but it can break down for models with many grouping factors. Bugs is more general (see Part 2B of this book) but can be slow with large datasets or complex models. In the meantime we need to start simple and build up gradually, a process during which we can also build understanding of the models being fit.

12.8 Predictions for new observations and new groups

Predictions for multilevel models can be more complicated than for classical regression because we can apply the model to existing groups or new groups. After a brief review of classical regression prediction, we explain in the context of the radon model.

Review of prediction for classical regression

In classical regression, prediction is simple: specify the predictor matrix $\tilde{X}$ for a set of new observations[5] and then compute the linear predictor $\tilde{X}\beta$, then simulate the predictive data:

- For linear regression, simulate independent normal errors $\tilde{\epsilon}_i$ with mean 0 and standard deviation σ, and compute $\tilde{y} = \tilde{X}\beta + \tilde{\epsilon}$; see Section 7.2.
- For logistic regression, simulate the predictive binary data: $\Pr(\tilde{y}_i) = \text{logit}^{-1}(\tilde{X}_i\beta)$ for each new data point i; see Section 7.4.
- With binomial logistic regression, specify the number of tries $\tilde{n}_i$ for each new unit i, and simulate $\tilde{y}_i$ from the binomial distribution with parameters $\tilde{n}_i$ and $\text{logit}^{-1}(\tilde{X}_i\beta)$; see Section 7.4.
- With Poisson regression, specify the exposures $\tilde{u}_i$ for the new units, and simulate $\tilde{y}_i \sim \text{Poisson}(\tilde{u}_i e^{\tilde{X}_i\beta})$ for each new i; see Section 7.4.

As discussed in Section 7.2, the estimation for a regression in R gives a set of n_{sims} simulation draws. Each of these is used to simulate the predictive data vector $\tilde{y}$, yielding a set of n_{sims} simulated predictions. For example, in the election forecasting example of Figure 7.5 on page 146:

R code

```
model.1 <- lm (vote.88 ~ vote.86 + party.88 + inc.88)
display (model.1)
n.sims <- 1000
sim.1 <- sim (model.1, n.sims)
beta.sim <- sim.1$beta
sigma.sim <- sim.1$sigma
n.tilde <- length (vote.88)
X.tilde <- cbind (rep(1,n.tilde), vote.88, party.90, inc.90)
y.tilde <- array (NA, c(n.sims, n.tilde))
for (s in 1:n.sims) {
  y.tilde[s,] <- rnorm (n.tilde, X.tilde%*%beta.sim[s,], sigma.sim[s])
}
```

This matrix of simulations can be used to get point predictions (for example, `median(y.tilde[,3])` gives the median estimate for $\tilde{y}_3$) or predictive intervals (for example, `quantile(y.tilde[,3],c(.025,.975))`) for individual data points or for more elaborate derived quantities, such as the predicted number of seats won by the Democrats in 1990 (see the end of Section 7.3). For many applications, the `predict()` function in R is a good way to quickly get point predictions and intervals (see page 48); here we emphasize the more elaborate simulation approach which allows inferences for arbitrary quantities.

[5] Predictions are more complicated for time-series models: even when parameters are fit by classical regression, predictions must be made sequentially. See Sections 8.4 and 24.2 for examples.

Prediction for a new observation in an existing group

We can make two sorts of predictions for the radon example: predicting the radon level for a new house within one of the counties in the dataset, and for a new house in a new county. We shall work with model (12.15) on page 266, with floor as an individual-level predictor and uranium as a group-level predictor

For example, suppose we wish to predict $\tilde{y}$, the log radon level for a house with no basement (thus, with radon measured on the first floor, so that $\tilde{x} = 1$) in Hennepin County ($j = 26$ of our Minnesota dataset). Conditional on the model parameters, the predicted value has a mean of $\alpha_{26} + \beta$ and a standard deviation of σ_y. That is,

$$\tilde{y}|\theta \sim \text{N}(\alpha_{26} + \beta\tilde{x}, \sigma_y^2),$$

where we are using θ to represent the entire vector of model parameters.

Given estimates of α, β, and σ_y, we can create a predictive simulation for $\tilde{y}$ using R code such as

R code

```
x.tilde <- 1
sigma.y.hat <- sigma.hat(M2)$sigma$data
coef.hat <- as.matrix(coef(M2)$county)[26,]
y.tilde <- rnorm (1, coef.hat %*% c(1, x.tilde, u[26]), sigma.y.hat)
```

More generally, we can create a vector of `n.sims` simulations to represent the predictive uncertainty in $\tilde{y}$:

R code

```
n.sims <- 1000
coef.hat <- as.matrix(coef(M2)$county)[26,]
y.tilde <- rnorm (1000, coef.hat %*% c(1, x.tilde, u[26]), sigma.y.hat)
```

Still more generally, we can add in the inferential uncertainty in the estimated parameters, α, β, and σ. For our purposes here, however, we shall ignore inferential uncertainty and just treat the parameters $\alpha, \beta, \sigma_y, \sigma_\alpha$ as if they were estimated perfectly from the data.[6] In that case, the computation gives us 1000 simulation draws of $\tilde{y}$, which we can summarize in various ways. For example,

R code

```
quantile (y.tilde, c(.25,.5,.75))
```

gives us a predictive median of 0.76 and a 50% predictive interval of $[0.26, 1.27]$. Exponentiating gives us a prediction on the original (unlogged) scale of $\exp(0.76) = 2.1$, with a 50% interval of $[1.3, 3.6]$.

For some applications we want the average, rather than the median, of the predictive distribution. For example, the expected risk from radon exposure is proportional to the predictive average or mean, which we can compute directly from the simulations:

R code

```
unlogged <- exp(y.tilde)
mean (unlogged)
```

In this example, the predictive mean is 2.9, which is a bit higher than the median of 2.1. This makes sense: on the unlogged scale, this predictive distribution is skewed to the right.

[6] One reason we picked Hennepin County ($j = 26$) for this example is that, with a sample size of 105, its average radon level is accurately estimated from the available data.

Prediction for a new observation in a new group

Now suppose we want to predict the radon level for a house, once again with no basement, but this time in a county not included in our analysis. We then must generate a new county-level error term, $\tilde{\alpha}$, which we sample from its $N(\gamma_0+\gamma_1\tilde{u}_j, \sigma_\alpha^2)$ distribution. We shall assume the new county has a uranium level equal to the average of the uranium levels in the observed counties:

R code
```
u.tilde <- mean (u)
```

grab the estimated $\gamma_0, \gamma_1, \sigma_\alpha$ from the fitted model:

R code
```
g.0.hat <- fixef(M2)["(Intercept)"]
g.1.hat <- fixef(M2)["u.full"]
sigma.a.hat <- sigma.hat(M2)$sigma$county
```

and simulate possible intercepts for the new county:

R code
```
a.tilde <- rnorm (n.sims, g.0.hat + g.1.hat*u.tilde, sigma.a.hat)
```

We can then simulate possible values of the radon level for the new house in this county:

R code
```
y.tilde <- rnorm (n.sims, a.tilde + b.hat*x.tilde, sigma.y.hat)
```

Each simulation draw of $\tilde{y}$ uses a different simulation of $\tilde{\alpha}$, thus propagating the uncertainty about the new county into the uncertainty about the new house in this county.

Comparison of within-group and between-group predictions. The resulting prediction will be more uncertain than for a house in a known county, since we have no information about $\tilde{\alpha}$. Indeed, the predictive 50% interval of this new $\tilde{y}$ is $[0.28, 1.34]$, which is slightly wider than the predictive interval of $[0.26, 1.27]$ for the new house in county 26. The interval is only slightly wider because the within-county variation in this particular example is much higher than the between-county variation.

More specifically, from the fitted model on page 266, the within-county (residual) standard deviation σ_y is estimated at 0.76, and the between-county standard deviation σ_α is estimated at 0.16. The log radon level for a new house in an already-measured county can then be measured to an accuracy of about ± 0.76. The log radon level for a new house in a new county can be predicted to an accuracy of about $\pm\sqrt{0.76^2 + 0.16^2} = \pm 0.78$. The ratio $0.78/0.76$ is 1.03, so we would expect the predictive interval for a new house in a new county to be about 3% wider than for a new house in an already-measured county. The change in interval width is small here because the unexplained between-county variance is so small in this dataset.

For another example, the 50% interval for the log radon level of a house with no basement in county 2 is $[0.28, 1.30]$, which is centered in a different place but also is narrower than the predictive interval for a new county.

Nonlinear predictions

Section 7.3 illustrated the use of simulation for nonlinear predictions from classical regression. We can perform similar calculations in multilevel models. For example, suppose we are interested in the average radon level among all the houses in Hennepin County ($j = 26$). We can perform this inference using poststratification, first estimating the average radon level of the houses with and without basements in the county, then weighting these by the proportion of houses in the county that have

basements. We can look up this proportion from other data sources on homes, or we can estimate it from the available sample data.

For our purposes here, we shall assume that 90% of all the houses in Hennepin County have basements. The average radon level of all the houses in the county is then 0.1 times the average for the houses in Hennepin County without basements, plus 0.9 times the average for those with basements. To simulate in R:

R code
```
y.tilde.basement <- rnorm (n.sims, a.hat[26], sigma.y.hat)
y.tilde.nobasement <- rnorm (n.sims, a.hat[26] + b.hat, sigma.y.hat)
```

We then compute the estimated mean for 1000 houses of each type in the county (first exponentiating since our model was on the log scale):

R code
```
mean.radon.basement <- mean (exp (y.tilde.basement))
mean.radon.nobasement <- mean (exp (y.tilde.nobasement))
```

and finally poststratify given the proportion of houses of each type in the county:

R code
```
mean.radon <- .9*mean.radon.basement + .1*mean.radon.basement
```

In Section 16.6 we return to the topic of predictions, using simulations from Bugs to capture the uncertainty in parameter estimates and then propagating inferential uncertainty into the predictions, rather than simply using point estimates `a.hat`, `b.hat`, and so forth.

12.9 How many groups and how many observations per group are needed to fit a multilevel model?

Advice is sometimes given that multilevel models can only be used if the number of groups is higher than some threshold, or if there is some minimum number of observations per groups. Such advice is misguided. Multilevel modeling includes classical regression as a limiting case (complete pooling when group-level variances are zero, no pooling when group-level variances are large). When sample sizes are small, the key concern with multilevel modeling is the estimation of variance parameters, but it should still work at least as well as classical regression.

How many groups?

When J, the number of groups, is small, it is difficult to estimate the between-group variation and, as a result, multilevel modeling often adds little in such situations, beyond classical no-pooling models. The difficulty of estimating variance parameters is a technical issue to which we return in Section 19.6; to simplify, when σ_α cannot be estimated well, it tends to be overestimated, and so the partially pooled estimates are close to no pooling (this is what happens when σ_α has a high value in (12.16) on page 269).

At the same time, multilevel modeling should not do any worse than no-pooling regression and sometimes can be easier to interpret, for example because one can include indicators for all J groups rather than have to select one group as a baseline category.

One or two groups

With only one or two groups, however, multilevel modeling reduces to classical regression (unless "prior information" is explicitly included in the model; see Section 18.3). Here we usually express the model in classical form (for example, including

a single predictor for `female`, rather than a multilevel model for the two levels of the `sex` factor).

Even with only one or two groups in the data, however, multilevel models can be useful for making predictions about new groups. See also Sections 21.2–22.5 for further connections between classical and multilevel models, and Section 22.6 for hierarchical models for improving estimates of variance parameters in settings with many grouping factors but few levels per factor.

How many observations per group?

Even two observations per group is enough to fit a multilevel model. It is even acceptable to have one observation in many of the groups. When groups have few observations, their α_j's won't be estimated precisely, but they can still provide partial information that allows estimation of the coefficients and variance parameters of the individual- and group-level regressions.

Larger datasets and more complex models

As more data arise, it makes sense to add parameters to a model. For example, consider a simple medical study, then separate estimates for men and women, other demographic breakdowns, different regions of the country, states, smaller geographic areas, interactions between demographic and geographic categories, and so forth. As more data become available it makes sense to estimate more. These complexities are latent everywhere, but in small datasets it is not possible to learn so much, and it is not necessarily worth the effort to fit a complex model when the resulting uncertainties will be so large.

12.10 Bibliographic note

Multilevel models have been used for decades in agriculture (Henderson, 1950, 1984, Henderson et al., 1959, Robinson, 1991) and educational statistics (Novick et al., 1972, 1973, Bock, 1989), where it is natural to model animals in groups and students in classrooms. More recently, multilevel models have become popular in many social sciences and have been reviewed in books by Longford (1993), Goldstein (1995), Kreft and De Leeuw (1998), Snijders and Bosker (1999), Verbeke and Molenberghs (2000), Leyland and Goldstein (2001), Hox (2002), and Raudenbush and Bryk (2002). We do not attempt to trace here the many applications of multilevel models in various scientific fields.

It might also be useful to read up on Bayesian inference to understand the theoretical background behind multilevel models.[7] Box and Tiao (1973) is a classic reference that focuses on linear models. It predates modern computational methods but might be useful for understanding the fundamentals. Gelman et al. (2003) and Carlin and Louis (2000) cover applied Bayesian inference including the basics of multilevel modeling, with detailed discussions of computational algorithms. Berger

[7] As we discuss in Section 18.3, multilevel inferences can be formulated non-Bayesianly; however, understanding the Bayesian derivations should help with the other approaches too. All multilevel models are Bayesian in the sense of assigning probability distributions to the varying regression coefficients. The distinction between Bayesian and non-Bayesian multilevel models arises only for the question of modeling the other parameters—the nonvarying coefficients and the variance parameters—and this is typically a less important issue, especially when the number of groups is large.

(1985) and Bernardo and Smith (1994) cover Bayesian inference from two different theoretical perspectives.

The R function `lmer()` is described by Bates (2005a, b) and was developed from the linear and nonlinear mixed effects software described in Pinheiro and Bates (2000).

Multilevel modeling used to be controversial in statistics; see, for example, the discussions of the papers by Lindley and Smith (1972) and Rubin (1980) for some sense of the controversy.

The Minnesota radon data were analyzed by Price, Nero, and Gelman (1996); see also Price and Gelman (2004) for more on home radon modeling.

Statistical researchers have studied partial pooling in many ways; see James and Stein (1960), Efron and Morris (1979), DuMouchel and Harris (1983), Morris (1983), and Stigler (1983). Louis (1984), Shen and Louis (1998), Louis and Shen (1999), and Gelman and Price (1999) discuss some difficulties in the interpretation of partially pooled estimates. Zaslavsky (1993) discusses adjustments for undercount in the U.S. Census from a partial-pooling perspective. Normand, Glickman, and Gatsonis (1997) discuss the use of multilevel models for evaluating health-care providers.

12.11 Exercises

1. Using data of your own that are appropriate for a multilevel model, write the model in the five ways discussed in Section 12.5.
2. Continuing with the analysis of the CD4 data from Exercise 11.4:
   (a) Write a model predicting CD4 percentage as a function of time with varying intercepts across children. Fit using `lmer()` and interpret the coefficient for time.
   (b) Extend the model in (a) to include child-level predictors (that is, group-level predictors) for treatment and age at baseline. Fit using `lmer()` and interpret the coefficients on time, treatment, and age at baseline.
   (c) Investigate the change in partial pooling from (a) to (b) both graphically and numerically.
   (d) Compare results in (b) to those obtained in part (c).
3. Predictions for new observations and new groups:
   (a) Use the model fit from Exercise 12.2(b) to generate simulation of predicted CD4 percentages for each child in the dataset at a hypothetical next time point.
   (b) Use the same model fit to generate simulations of CD4 percentages at each of the time periods for a new child who was 4 years old at baseline.
4. Posterior predictive checking: continuing the previous exercise, use the fitted model from Exercise 12.2(b) to simulate a new dataset of CD4 percentages (with the same sample size and ages of the original dataset) for the final time point of the study, and record the average CD4 percentage in this sample. Repeat this process 1000 times and compare the simulated distribution to the observed CD4 percentage at the final time point for the actual data.
5. Using the radon data, include county sample size as a group-level predictor and write the varying-intercept model. Fit this model using `lmer()`.
6. Return to the beauty and teaching evaluations introduced in Exercise 3.5 and 4.8.

(a) Write a varying-intercept model for these data with no group-level predictors. Fit this model using `lmer()` and interpret the results.

(b) Write a varying-intercept model that you would like to fit including three group-level predictors. Fit this model using `lmer()` and interpret the results.

(c) How does the variation in average ratings across instructors compare to the variation in ratings across evaluators for the same instructor?

7. This exercise will use the data you found for Exercise 4.7. This time, rather than repeating the same analysis across each year, or country (or whatever group the data varies across), fit a multilevel model using `lmer()` instead. Compare the results to those obtained in your earlier analysis.

8. Simulate data (outcome, individual-level predictor, group indicator, and group-level predictor) that would be appropriate for a multilevel model. See how partial pooling changes as you vary the sample size in each group and the number of groups.

9. Number of observations and number of groups:

(a) Take a simple random sample of one-fifth of the radon data. (You can create this subset using the `sample()` function in R.) Fit the varying-intercept model with floor as an individual-level predictor and log uranium as a county-level predictor, and compare your inferences to what was obtained by fitting the model to the entire dataset. (Compare inferences for the individual- and group-level standard deviations, the slopes for floor and log uranium, the average intercept, and the county-level intercepts.)

(b) Repeat step (a) a few times, with a different random sample each time, and summarize how the estimates vary.

(c) Repeat step (a), but this time taking a cluster sample: a random sample of one-fifth of the counties, but then all the houses within each sampled county.

CHAPTER 13

Multilevel linear models: varying slopes, non-nested models, and other complexities

This chapter considers some generalizations of the basic multilevel regression. Models in which slopes and intercepts can vary by group (for example, $y_i = \alpha_{j[i]} + \beta_{j[i]}x_i + \cdots$, where α and β both vary by group j; see Figure 11.1c on page 238) can also be interpreted as interactions of the group index with individual-level predictors.

Another direction is non-nested models, in which a given dataset can be structured into groups in more than one way. For example, persons in a national survey can be divided by demographics or by states. Responses in a psychological experiment might be classified by person (experimental subject), experimental condition, and time.

The chapter concludes with some examples of models with nonexchangeable multivariate structures. We continue with generalized linear models in Chapters 14–15 and discuss how to fit all these models in Chapters 16–19.

13.1 Varying intercepts and slopes

The next step in multilevel modeling is to allow more than one regression coefficient to vary by group. We shall illustrate with the radon model from the previous chapter, which is relatively simple because it only has a single individual-level predictor, x (the indicator for whether the measurement was taken on the first floor).

We begin with a varying-intercept, varying-slope model including x but without the county-level uranium predictor; thus,

$$\begin{aligned} y_i &\sim \text{N}(\alpha_{j[i]} + \beta_{j[i]}x_i,\, \sigma_y^2), \text{ for } i = 1, \ldots, n \\ \begin{pmatrix} \alpha_j \\ \beta_j \end{pmatrix} &\sim \text{N}\left(\begin{pmatrix} \mu_\alpha \\ \mu_\beta \end{pmatrix}, \begin{pmatrix} \sigma_\alpha^2 & \rho\sigma_\alpha\sigma_\beta \\ \rho\sigma_\alpha\sigma_\beta & \sigma_\beta^2 \end{pmatrix} \right), \text{ for } j = 1, \ldots, J, \end{aligned} \tag{13.1}$$

with variation in the α_j's and the β_j's and also a between-group correlation parameter ρ. In R:

R code

```
M3 <- lmer (y ~ x + (1 + x | county))
display (M3)
```

which yields

R output

```
lmer(formula = y ~ x + (1 + x | county))
            coef.est coef.se
(Intercept)  1.46     0.05
x           -0.68     0.09
Error terms:
 Groups   Name        Std.Dev. Corr
 county   (Intercept) 0.35
          x           0.34     -0.34
```