

Multilevel modeling is most necessary when at least some groups need to borrow information from the others. When σ_α is small and groups are similar.

If σ_α is large, groups vary a lot and multilevel is like no-pooling.

We don't want to force all groups to be alike. Partial pooling works well with unequal sample sizes (and precisions)

Stat literature on random effects terms $\hat{\alpha}_j$ a prediction, not an estimate. Do not ask if we need α_j in the model.

Radon example (ignoring floor and uranium)

County	Estimate	SE(estimate)	Est/SE
AITKIN	1.31 -0.2451	0.245	-1
ANOKA	1.31 -0.425	0.104	-4
BECKER	1.31 -0.082	0.257	-1/3

Anoka county effect is not "highly significant" (even relative to Aitken). We would not drop one and keep another. We could remove all of them if we thought county variance was ignorable. Don't use small individual p-values as a reason to exclude variables. Biggest constraints: fitting and understanding complex models.

Start Simple and Add Complexity

- Complete Pooling – one model ignoring group effects. Can include group-level predictors.
- No pooling – a model with group effects, but no group-level predictors.
- Separate models – one for each level $j = 1, \dots, J$ using no group level predictors (may be limited by sample sizes, n_j).
- Two-stage analysis using no-pooling or separate models to get group level estimates. Then treat these as data in a group-level model.

Group-level predictors reduce the variance between groups, σ_α^2 giving more precise estimates of α_j .

Reminders from classical regression:

- Use the predict function in R with `newdata=list(x=x.new)`, `se.fit=T` to get estimated \hat{y}_{new} and its SE. To predict for a new observation $SE_{prediction} = \sqrt{s^2 + SE(\hat{y}_{new})^2}$
- Use the sim function in arm package to generate random $\tilde{s}^2 \sim \text{Scaled-Inv-}\chi^2$ and $\tilde{\beta} \sim N(\hat{\beta}, \tilde{s}^2(\mathbf{X}^T\mathbf{X})^{-1})$. Then new predicted response is $\tilde{y}_{new} \sim N(\mathbf{X}_{new}\tilde{\beta}, \tilde{s}^2)$
- Logistic regression: generate random $\mathbf{X}_{new}\tilde{\beta}$ as above, convert to probability with $\tilde{p}_i = \text{logit}^{-1}(\mathbf{X}_{new}\tilde{\beta})$ and generate random $\text{Bin}(n_i, \tilde{p}_i)$.
- Poisson: $\tilde{y}_{new} \sim \text{Poisson}(u_{new}e^{\mathbf{X}_{new}\tilde{\beta}})$ where u is exposure.

R functions do not work with derived quantities, but sim does.

```
> elect90.fit <- lm(DemRatio ~ DemRatio88*party + incumbent,
  elect2)
> sim.1 <- sim(elect90.fit, 1000)
> sigma.sim <- sim.1@sigma
> X.tilde <- model.matrix(elect90.fit)
> n.tilde <- nrow(X.tilde)
> y.tilde <- X.tilde %*% t(sim.1@coef) + rnorm(n.tilde*1000,
  0, rep(sigma.sim, each=n.tilde))
> quantile(y.tilde[,3], c(0.025, .25, .5, .75, .975))
```

2.5%	25%	50%	75%	97.5%
0.251	0.404	0.547	0.662	0.805

Predict log radon level for the first floor of a new house in Hennepin county. (county 26)

Assume all coefficients are estimated without error.

```
> sigma.y.hat <- sigma.hat(lmer.fit2)$sigma$data
> beta.hat <- as.matrix(coef(lmer.fit2)$county)[26,]
> y.tilde.floor1 <- rnorm(1000, beta.hat %*% c(1, 1, 0.908),
  sigma.y.hat) # 0.908 is mean(u) in county 26
> exp(quantile(y.tilde.floor1, c(0.025, .5, .975)))
```

2.5%	50%	97.5%
0.903	4.251	17.594

```
> ## Simulate a non-linear function -- radon an average
  Hennepin house
> y.tilde.floor0 <- rnorm(1000, beta.hat %*% c(1, 0, 0.908),
  sigma.y.hat)
> summary(.9 * exp(y.tilde.floor0) + .1 * exp(
  y.tilde.floor1))
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
0.6	4.8	7.7	10.2	12.7	74.3

Now we don't know α_j , must draw it from $N(\gamma_0 + \gamma_1 u_{\text{new}}, \sigma_\alpha^2)$

```
> sigma.a.hat <- sigma.hat(lmer.fit2)$sigma$county
> gamma.hat <- fixef(lmer.fit2)[-2] ## leave out floor
  effect
> a.tilde <- rnorm(1000, gamma.hat[1] + gamma.hat[2]*mean(
  minn$u), sigma.a.hat)
> y.tilde <- rnorm(1000, a.tilde + beta.hat[2]*1, sigma.y.hat)
  # U already accounted for in a.tilde
> quantile(y.tilde, c(0.025, .25, .5, .75, .975))
```

2.5%	25%	50%	75%	97.5%
-0.695	0.217	0.732	1.224	2.256

```
> exp(quantile(y.tilde, c(0.025, .5, .975))) ## back to pC/l
```

2.5%	50%	97.5%
0.499	2.080	9.546

With a small number of groups, J , it's hard to estimate σ_α . Tends to get over-estimated, making multilevel like no-pooling.

Need at least two observations in some groups to estimate σ_y .