# Multilevel Models

# Motivation

- Many types of data have natural hierarchies
  - counties in states in regions
  - years in decades in centuries
- MLMs control for unobserved confounders between groups
- MLMs estimate separate coefficients across groups
- MLMs model each level of a hierarchy with separate predictors.

# What MLMs Aren't

- Not new a model with arcane functional form
- Not complete-pooling

$$y_i = \alpha + \beta X_i + \epsilon$$

- Not No-pooling

$$y_{ij} = \alpha_j + \beta_j X_{ij} + \epsilon_j$$

# Overview

- What can we model?
- How is it modeled?
- How is it estimated?
- Has it converged?
- How can we interpret the results?

# What can we model?

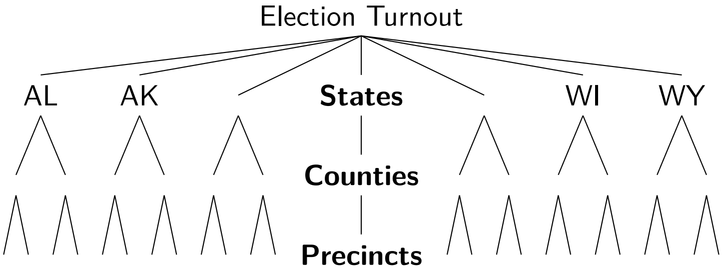
- ... any organized hierarchy of groups.
- Separate slopes or intercepts (or both) for
- non-nested (time-series)
  - e.g. Republican presidential vote shares from 1900 to 2016
- nested (single hierarchy)
  - e.g. precints in counties in states
- non-nested groups (multiple, overlapping hierarchies)
  - legislators nested within parties and across chambers

# What can we model? Non-nested Models

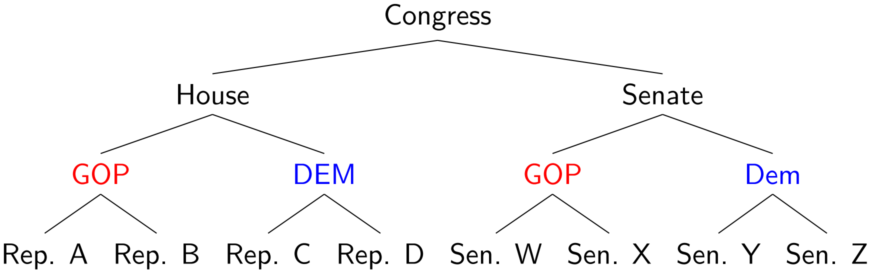
GOP Voteshare

1900 1904 **Year** 2012 2016

# What can we model? Nested Models



# What can we model? Non-nested Hierarchical Models



# How do we model Non-nested Models?

- Let's say we're modeling person $j$'s vote choice, $y_{jt}$, in election $t$:

$$\Pr\left(y_{jt} = 1\right) = \text{logit}^{-1}\left(\alpha_j + \beta_1 \text{Party} + \beta_2 y_{j,t-1}\right)$$

# How do we model Nested Models?

Let's say we're modeling annual turnout, $y_i$, in precincts, $k$, within states, $j$ (ignoring time-series considerations)

$$
\begin{aligned}
y_{ijk} &\sim N\left(\alpha_j + \beta_1 \text{Pres. Approval}, \sigma_y^2\right) \\
\alpha_{jk} &\sim N\left(\gamma_k + \gamma_1 \text{Voter ID}_j, \sigma_\alpha^2\right) \\
\gamma_k &\sim N\left(\lambda_0 + \lambda_1 \text{Rain}_k, \sigma_\gamma^2\right)
\end{aligned}
$$

# How do we model Non-nested Hierarchical Models?

Let's say we're modeling legislators' votes to change healthcare policy, $y_i$.

Legislators belong to both a party, $j$, and a chamber, $k$

$$
\begin{aligned}
y_{ijk} &\sim N\left(\alpha_k + \beta_1 \text{Ideology} + \gamma_j \text{Obama Pres.}, \sigma_y^2\right) \\
\alpha_k &\sim N\left(\eta_0, \sigma_\alpha^2\right) \\
\gamma_j &\sim N\left(\lambda_0, \sigma_\gamma^2\right)
\end{aligned}
$$

(This example's trivial since both are binary)

(For fun, re-consider this case for state legislators)

# How do we estimate it?

- ... with Bayesian inference.
- Parameters are estimated via Gibbs Sampling
- Iterate over the posterior densities defined for each equation
- R has `library(rstan)` package for simple models
- Common softwares: BUGS, JAGS, STAN

# How do we estimate it? LME4

- Linear Mixed Effects models
- `lme4::lmer` for linear regression
- `lme4::glmer` for generalized linear models
- `lme4::nlmer` for non-linear models
- Convenient, but inflexible
- Estimates parameters by maximizing restricted maximum likelihood (REML)

# How do we estimate it? LME4

```
library(foreign)
nes <- read.dta("nes.dta", convert.factors = FALSE)
# subset for year 2000
nes00 <- nes[nes$year == 2000, ]
# subset relevant variables
nes00 <- nes00[ , c("partyid7", "real_ideo", "state")]
# remove NAs from data
# Note: DO NOT DELETE BY CASEWISE MISSINGNESS
# Think very carefully about your type of missingness
#    and if imputation is appropriate
nes00 <- nes00[complete.cases(nes00), ]
# clean state variable
nes00$state <- match(nes00$state, unique(nes00$state))
```

```r
# load library to fit lmers
library(lme4)
# Model I'd Like to Fit
Model <- lmer(real_ideo ~ 1 + partyid7 + (1 | state),
              data = nes00)
```

```r
summary(Model, correlation = F)
```

```
...
Random effects:
 Groups   Name        Variance Std.Dev.
 state    (Intercept) 0.09751  0.3123
 Residual             1.34496  1.1597
Number of obs: 571, groups:  state, 46

Fixed effects:
            Estimate Std. Error t value
(Intercept)  3.01811    0.11658   25.89
partyid7     0.34105    0.02289   14.90
...
```

```r
# for random effects,
ranef(Model)
# ... fixed effects,
fixef(Model)
# ... and their SEs
library(arm)
se.ranef(Model)
se.fixef(Model)
# also the overall intercept and slope for all observations
coef(Model)
```

# How do we estimate it? STAN

- C++ library for Bayesian inference
- Uses No-U-Turn MCMC sampling to evaluate posteriors
- Run in `R` with `library(rstan)`
- Note: Normal(mean, *std. dev.*)

# How do we estimate it? STAN

```r
library(rstan)

# Model I'd Like to Fit
STANcode <- '
data {
  int<lower=1> N; // number of observations
  int<lower=1> J; // number of states
  int<lower=1> STATE[N]; // state indicators
  vector[N] PID; // party id
  vector[N] IDEO; // ideology
}
'
```

```
'
parameters {
  vector[J] alpha; // vector, length states, for intercepts
  real beta; // coef. on ideology
  real mu_alpha; // mean for intercepts
  real<lower=0> sigma_y; // resid. sd. for sample-level
  real<lower=0> sigma_alpha; // resid. sd. for group-level
}
transformed parameters {
  vector[N] y_hat; // vector, length obs., for fitted y
  y_hat = alpha[STATE] + PID * beta; // est. fitted y
}
'
```

```
'
model {
  sigma_alpha ~ inv_gamma(0.001, 0.001); // samp. group sd.
  alpha ~ normal(mu_alpha, sigma_alpha); // est. group model

  beta ~ normal(0, 1); // samp. slope

  sigma_y ~ inv_gamma(0.001, 0.001); // samp. sample-level sd.
  IDEO ~ normal(y_hat, sigma_y); // est. sample-level model
}
'
```

```
'
parameters {
  ...
  real beta[2]; // coef. on ideology and educ
  real gamma; // inter. group
  ...
}
transformed parameters {
  ...
  vector[J] alpha_hat; // vector, length state, for fitted a
  y_hat = alpha_hat + PID * beta[1]; // est. fitted y
  alpha_hat = gamma + EDUC * beta[2]; // est. fitted a
}
model {
  ...
  gamma ~ normal(0, 1); samp. group inter.
  ...
}
'
```

```r
# compile STAN model
STANmodel <- stan_model(model_code = STANcode)

# estimate
STANfit <- sampling(STANmodel,
                    data = list(PID = nes00$partyid7,
                                IDEO = nes00$real_ideo,
                                STATE = nes00$state,
                                N = nrow(nes00),
                                J = max(nes00$state)),
                    iter = 5000,
                    warmup = 1000,
                    chains = 5,
                    cores = 3)
```

```r
print(STANfit, pars=c("beta", "sigma_y", "sigma_alpha",
                      "alpha[1]", "alpha[2]", "alpha[3]"),
      probs=c(.025,.975))
```

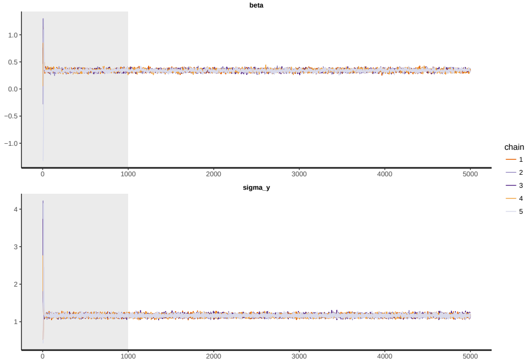
```
...
            mean se_mean   sd 2.5% 97.5% n_eff Rhat
beta        0.34       0 0.02 0.30  0.39  1922 1.00
sigma_y     1.16       0 0.04 1.10  1.24 20000 1.00
sigma_alpha 0.30       0 0.09 0.11  0.48   495 1.01
alpha[1]    2.87       0 0.30 2.25  3.45 20000 1.00
alpha[2]    2.65       0 0.23 2.18  3.08 20000 1.00
alpha[3]    3.24       0 0.23 2.81  3.71 20000 1.00
...
```

# Has it converged?

- Time & iterations to convergence depends on the model
- Convergence occurs when all chains are sampling from the same posterior density
- $\hat{R}$ lets us assess this numerically
- $$\hat{R} = \frac{\text{Var(Between)} + \text{Var(Within)}}{\text{Var(Within)}}$$
- Traceplots let us assess this visually
- Marginal density plots
- $$ESS = \frac{n}{1 + 2\sum_{k=1}^{\infty} \rho(k)}$$

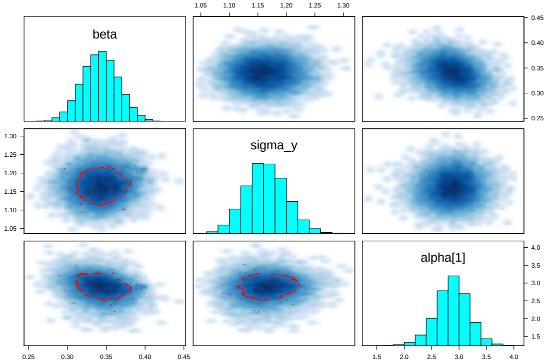
# Has it converged?

```r
rstan::traceplot(STANfit,
                 pars = c("beta", "sigma_y"),
                 inc_warmup = TRUE,
                 nrow = 2)
```



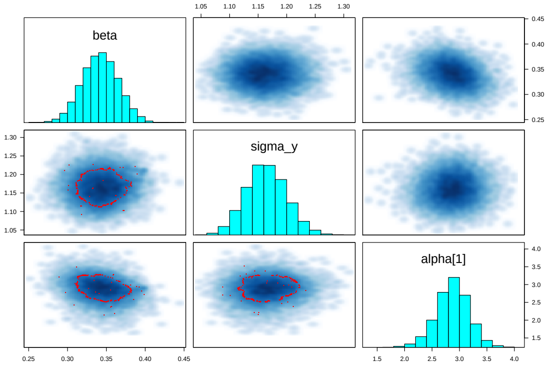
# Does the sampling look skewed?

```
pairs(STANfit,
      pars = c("beta", "sigma_y", "alpha[1]"),
      las = 1)
```



# Does the sampling look skewed?

- Distributions above/below the diagonal should be mirrored
- Red points indicate divergent transitions



# How can we interpret the results?

- Result interpretation depends on the model

- Interpret the following as you would for single-level models
  - intercepts
  - slopes
  - corresponding SEs

- Parameters unique to MLM are
  - $\sigma_y^2$ (within-group variation)
  - measurement error, natural variation, between-unit variation
  - $\sigma_{(\cdot)}^2$ (between-group variation)
  - variation between groups that is not explained by group-level predictors
  - $\rho = \frac{\sigma_{(\cdot)}^2}{\sigma_{(\cdot)}^2 + \sigma_y^2}$ (intraclass correlation coefficient)

# How can we interpret the results *well*?

- Substantive explanations of intercepts, slopes, and variances are easy
- But satisfying explanations for these quantities across n groups is hard
- Graphics are helpful
- Contrast averages with outliers, or salient cases

# Plotting varying parameters

```r
plot(STANfit, pars = c("alpha"), las = 1)
```