Hierarchical Models I

Ben Goodrich

StanCon Helsinki: August 29, 2018

Obligatory Disclosure

- Ben is an employee of Columbia University, which has received several research grants to develop Stan
- Ben is also a manager of GG Statistics LLC, which utilizes Stan for business purposes
- According to Columbia University policy, any such employee who has any equity stake in, a title (such as officer or director) with, or is expected to earn at least $\$5,000.00$ per year from a private company is required to disclose these facts in presentations

Goals for the First Session

- Think about conditional distributions, the building blocks for hierarchical models
- Practice writing functions in the Stan language to draw from the prior predictive distribution
- Write simple Stan programs where some parameters are functions of other parameters
- Estimate a hierarchical model using `rstanarm::stan_glmer`

Hierarchical Data Generating Processes: Bowling

- How to model what person i does on the jth bowling frame?

Hierarchical Data Generating Processes: Bowling

- How to model what person i does on the jth bowling frame?
- You would need (at least) two probability distributions:
  1. Probability of knocking down $x_1 \in \{0, 1, \ldots, 10\}$ pins on the first roll
  2. Probability of knocking down $x_2 \in \{0, 1, \ldots, 10 - x_1\}$ pins on the second roll, given that x_1 pins were knocked down on the first roll

```
x_1 <- sample(0:10, size = 1)
pins_left <- 10 - x_1
x_2 <- sample(0:pins_left, size = 1)
x_1 + x_2

## [1] 4
```

- All Stan does is draw from a conditional probability distribution

Hierarchical Data Generating Processes: IV

A generative model for an instrumental variable (IV) design is

$$\begin{aligned} \sigma_1 &\sim \text{Exponential}(r_1) \\ \text{Priors: } \sigma_2 &\sim \text{Exponential}(r_2) \\ \rho &\sim \text{Uniform}(-1,1) \\ \text{Errors: } \begin{bmatrix} \nu_i \\ \varepsilon_i \end{bmatrix} &\sim \mathcal{N}_2\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}\right) \forall i \end{aligned}$$

$$\text{Priors: } \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \end{bmatrix} \sim \mathcal{N}_3(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1) \qquad \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix} \sim \mathcal{N}_3(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$$

$$\begin{aligned} \text{1st stage: } t_i &\equiv \alpha_0 + \alpha_1 x_i + \alpha_2 z_i + \nu_i \forall i \\ \text{2nd stage: } y_i &\equiv \beta_0 + \beta_1 x_i + \beta_2 t_i + \varepsilon_i \forall i \end{aligned}$$

Coefficients Depending on Other Coefficients

Write a simple Stan program where the coefficient on age is a linear function of the person's income, which both affect the probability of voting in a logit model, starting with

```
data {
  int<lower=1> N;
  vector[N] age;
  vector[N] income;
  int<lower=0,upper=1> vote[N]; // outcome
}
parameters {
  vector[2] lambda;    // intercept / slope for age's effect
  vector[2] beta;      // intercept / slope for log-odds
}
```

Coefficients Depending on Other Coefficients

Write a simple Stan program where the coefficient on age is a linear function of the person's income, which both affect the probability of voting in a logit model, starting with

```
data {
  int<lower=1> N;
  vector[N] age;
  vector[N] income;
  int<lower=0,upper=1> vote[N]; // outcome
}
parameters {
  vector[2] lambda;    // intercept / slope for age's effect
  vector[2] beta;      // intercept / slope for log-odds
}
model {
  vector[N] beta_age = lambda[1] + lambda[2] * income;
  vector[N] eta = beta[1] + beta[2] * income
                + beta_age .* age;
  target += bernoulli_logit_lpmf(vote | eta);
}  // priors on lambda, beta, and sigma
```

Relation to Interaction Terms in R

- If

$$\begin{aligned} \eta_i &= \beta_1 + \beta_2 \times \text{Income}_i + \beta_{3i} \times \text{Age}_i \\ \beta_{3i} &= \lambda_1 + \lambda_2 \times \text{Income}_i \end{aligned}$$

  then by substituting and distributing:

$$\begin{aligned} \eta_i &= \beta_1 + \beta_2 \times \text{Income}_i + (\lambda_1 + \lambda_2 \times \text{Income}_i) \times \text{Age}_i \\ &= \beta_1 + \beta_2 \times \text{Income}_i + \lambda_1 \times \text{Age}_i + \lambda_2 \times \text{Income}_i \times \text{Age}_i \end{aligned}$$

  and β_1, β_2, λ_1, and λ_2 can be estimated (unregularized) via

```
glm(vote ~ income + age + income:age, family = binomial)
```

- Stan version is easier to interpret; R version is quick
- Many hierarchical models are just interactions w/ group indicators

Coefficients Depending on Other Coefficients Again

Write a Stan program where the coefficient on age is a **noisy** linear function of the person's income with standard deviation σ, starting with

```
data {
  int<lower=1> N; vector[N] age;
  vector[N] income; int<lower=0,upper=1> vote[N];
}
parameters {
  vector[2] lambda;      // intercept / slope for age's effect
  vector[N] noise;       // error in effect of age
  real<lower=0> sigma;   // sd of error in beta_age
  vector[2] beta;        // intercept / slope for log-odds
}
```

Coefficients Depending on Other Coefficients Again

Write a Stan program where the coefficient on age is a **noisy** linear function of the person's income with standard deviation σ, starting with

```
data {
  int<lower=1> N; vector[N] age;
  vector[N] income; int<lower=0,upper=1> vote[N];
}
parameters {
  vector[2] lambda;       // intercept / slope for age's effect
  vector[N] noise;        // error in effect of age
  real<lower=0> sigma;    // sd of error in beta_age
  vector[2] beta;         // intercept / slope for log-odds
}
model {
  vector[N] beta_age = lambda[1] + lambda[2] * income
                     + sigma * noise; // non-centering
  vector[N] eta = beta[1] + beta[2] * income
                + beta_age .* age;
  target += bernoulli_logit_lpmf(vote | eta);
  target += normal_lpdf(noise | 0, 1);
}  // priors on lambda, sigma, and beta
```

Cluster Sampling Designs

- Classic example of cluster sampling:
  1. Randomly draw J schools from the population of schools
  2. For each selected school, randomly draw N_j students
  3. Collect data on these $N = \sum_{j=1}^{J} N_j$ students
- If one tried to replicate this study, both the schools and the students would be different than in the original study

$$\begin{aligned}
\tau &\sim \text{Exponential}\left(r_\tau\right) \\
\alpha_j &\sim \mathcal{N}\left(0, \tau\right) \forall j \\
\beta &\sim \mathcal{N}\left(\mu_\beta, \sigma_\beta\right) \\
\sigma_\varepsilon &\sim \text{Exponential}\left(r_\sigma\right) \\
\varepsilon_{ij} &\sim \mathcal{N}\left(0, \sigma_\varepsilon\right) \\
y_{ij} &\equiv \alpha_j + \beta \times \text{class_size}_{ij} + \varepsilon_{ij} \forall i, j
\end{aligned}$$

Write a Stan Function to Draw from this DGP

```
functions {
  vector cluster_DGP_rng(int J, int[] N, vector class_size,
                         real r_tau, real r_sigma,
                         real mu_beta, real sigma_beta) {
```

$$
\begin{aligned}
\tau &\sim \text{Exponential}\left(r_\tau\right) \\
\alpha_j &\sim \mathcal{N}\left(0, \tau\right) \forall j \\
\beta &\sim \mathcal{N}\left(\mu_\beta, \sigma_\beta\right) \\
\sigma_\varepsilon &\sim \text{Exponential}\left(r_\sigma\right) \\
\varepsilon_{ij} &\sim \mathcal{N}\left(0, \sigma_\varepsilon\right) \\
y_{ij} &\equiv \alpha_j + \beta \times \mathsf{class\_size}_{ij} + \varepsilon_{ij} \forall i, j
\end{aligned}
$$

Stan Function to Draw from this DGP

```
vector cluster_DGP_rng(int J, int[] N, vector class_size,
                       real r_tau, real r_sigma,
                       real mu_beta, real sigma_beta) {
  real tau = exponential_rng(r_tau);
  real sigma = exponential_rng(r_sigma);
  real beta = normal_rng(mu_beta, sigma_beta);
  vector[sum(N)] y;
  int pos = 1;
  for (j in 1:J) {
    real alpha_j = normal_rng(0, tau);
    for (i in 1:N[j]) {
      real mu = alpha_j + beta * class_size[pos];
      real epsilon = normal_rng(0, sigma)
      y[pos] = mu + epsilon;
      pos += 1;
    }
  }
  return y;
}
```

Exposing Stan Functions to R

- If you put the previous function inside the functions block of an otherwise empty Stan program, you can export it to R

```
rstan::expose_stan_functions("schools.stan")
```

```
args(cluster_DGP_rng)

## function (J, N, class_size, r_tau, r_sigma, mu_beta, sigma_beta,
##     seed = 0L)
## NULL
```

- Now you can call the `cluster_DGP_rng` function with those arguments and get back one vector of prior predictions
- Doing so repeatedly is a good way to judge whether your priors make sense

Hierarchical Models

- A hierarchical model is one where a prior is specified on a parameter conditional on another unknown parameter
- Hierarchical models are often used in situations to allow parameters to vary by categorical group
- Suppose there are J groups & N_j observations in jth group
- Best way to think about such structures:
  - There is a likelihood contribution for the jth group
  - There are priors over how parameters vary across groups
  - There are priors on parameters common to all groups
- Relevant prior information pertains to how similar you believe the groups' data-generating processes to be

Table 2 from the **lme4** Vignette (frequentist)

Formula	Alternative	Meaning
`(1 | g)`	`1 + (1 | g)`	Random intercept with fixed mean.
`0 + offset(o) + (1 | g)`	`-1 + offset(o) + (1 | g)`	Random intercept with *a priori* means.
`(1 | g1/g2)`	`(1 | g1) + (1 | g1:g2)`	Intercept varying among `g1` and `g2` within `g1`.
`(1 | g1) + (1 | g2)`	`1 + (1 | g1) + (1 | g2)`	Intercept varying among `g1` and `g2`.
`x + (x | g)`	`1 + x + (1 + x | g)`	Correlated random intercept and slope.
`x + (x || g)`	`1 + x + (1 | g) + (0 + x | g)`	Uncorrelated random intercept and slope.

Table 2: Examples of the right-hand-sides of mixed-effects model formulas. The names of grouping factors are denoted `g`, `g1`, and `g2`, and covariates and *a priori* known offsets as `x`

- The **lme4** parser converts statements like `x + (x | g)` to a sparse matrix $\mathbf{Z}$ that interacts (some columns of) $\mathbf{X}$ with group-specific dummy variables, one for each level of `g`

Restatement of the Hierarchical Linear Model

- Generally, both intercepts and slopes can vary across groups
- Let $\boldsymbol{\beta}_j = \boldsymbol{\beta} + \mathbf{b}_j$ and $\mathbf{b}^\top = \begin{bmatrix} \mathbf{b}_1^\top & \mathbf{b}_2^\top & \cdots & \mathbf{b}_J^\top \end{bmatrix}$. Then:

$$\mathbf{y} = \overbrace{\underbrace{\mathbf{X}\boldsymbol{\beta}}_{\text{Frequentist } \boldsymbol{\mu}} + \mathbf{Z}\mathbf{b}}^{\text{Bayesian } \boldsymbol{\mu}} + \boldsymbol{\varepsilon} = \mathbf{X}\boldsymbol{\beta} + \underbrace{\mathbf{Z}\overbrace{\mathbf{L}(\boldsymbol{\theta})\mathbf{u}\sigma}^{\mathbf{b}} + \overbrace{\boldsymbol{\varepsilon}}^{\text{Bayesian error}}}_{\text{Frequentist error}}$$

where $\mathbf{L}(\boldsymbol{\theta})$ is a Cholesky factor of $\text{cov}(\mathbf{b}) = \boldsymbol{\Sigma}(\boldsymbol{\theta}) = \mathbf{L}(\boldsymbol{\theta})\mathbf{L}(\boldsymbol{\theta})^\top$

- Bayesians: $\mathbf{b} \sim \mathscr{N}(\mathbf{0}, \boldsymbol{\Sigma}(\boldsymbol{\theta}))$ and $\mathbf{y} \sim \mathscr{N}\left(\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{b}, \sigma^2\mathbf{I}\right)$
- For frequentists, each $\mathbf{b}_j$ is not a fixed "parameter" but rather a random variable that is part of the error term that gets integrated out to choose $\widehat{\boldsymbol{\beta}}, \widehat{\sigma}$, and $\boldsymbol{\Sigma}\left(\widehat{\boldsymbol{\theta}}\right)$ to maximize a multivariate normal likelihood with mean $\mathbf{X}\boldsymbol{\beta}$ and covariance matrix $\sigma^2\mathbf{Z}\boldsymbol{\Sigma}(\boldsymbol{\theta})\mathbf{Z}^\top$
- Technically, $\widehat{\mathbf{b}}_j$ is not "estimated" but rather "predicted" from the residuals $\mathbf{e} = \mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}}$ by subsequently regressing $\mathbf{e}$ on $\mathbf{Z}$

Bayesian Implementations with **lme4** / **mgcv** Syntax

- The **rstanarm** and **brms** packages accept **lme4** syntax
- Both also permit the same `s(...)` syntax as **mgcv** to use smooth, non-linear functions of parameters
- Add arguments for the priors on α, $\boldsymbol{\beta}$, σ, etc.
- Try **rstanarm** and / or **brms** first to make sure your data are amenable to a hierarchical model

Tadpole Example from McElreath, chapter 12

```
GH <- "https://raw.githubusercontent.com/"
FILE <- "rmcelreath/rethinking/master/data/reedfrogs.csv"
reedfrogs <- read.table(paste0(GH, FILE), sep = ";",
                        header = TRUE)
reedfrogs$tank <- as.factor(1:nrow(reedfrogs)) # groups
library(rstanarm); options(mc.cores = parallel::detectCores())
post <- stan_glmer(cbind(surv, density - surv) ~ size +
                   (1 | tank), data = reedfrogs,
                   family = binomial('logit'))
```

```
dim(as.matrix(post)) # raw draws from posterior distribution

## [1] 4000   51
```

Results of Tadpole Example from McElreath

```
## stan_glmer
##  family:       binomial [logit]
##  formula:      cbind(surv, density - surv) ~ size + (1 | tank)
##  observations: 48
## ------
##             Median MAD_SD
## (Intercept) 1.2    0.4
## sizesmall   0.4    0.5
##
## Error terms:
##  Groups Name        Std.Dev.
##  tank   (Intercept) 1.7
## Num. levels: tank 48
##
## Sample avg. posterior predictive distribution of y:
##          Median MAD_SD
## mean_PPD 16.3   0.4
##
## ------
## For info on the priors used see help('prior_summary.stanreg').
```

More Results of Tadpole Example from McElreath

```
fixef(post)

## (Intercept)   sizesmall
##   1.1524158   0.4427099

NROW(ranef(post)$tank)

## [1] 48

head(cbind(coef(post)$tank[,1],
           fixef(post)[1] + ranef(post)$tank))

##   coef(post)$tank[, 1] (Intercept)
## 1             2.018290    2.018290
## 2             2.914472    2.914472
## 3             0.946066    0.946066
## 4             2.899101    2.899101
## 5             1.758918    1.758918
## 6             1.731015    1.731015
```