Bayesian Multi-level Regression

Bayesian Modeling in brms

Michael D. Nunez

29-August-2022

Amsterdam Mathematical Psychology Laboratory
University of Amsterdam, The Netherlands

Sections

The data

Obtaining the real data set

```
# install.packages('curl')
library(curl)

# See https://github.com/mdnunez/encodingN200
pdmdat <- curl("https://tinyurl.com/PDMdataESCOP2022")
pdm <- read.csv(pdmdat)
colnames(pdm) <- c('N200_latencies', 'N200_amplitudes',
    'RT', 'accuracy', 'condition', 'EEG_session',
    'experiment', 'session', 'subject')
pdm <- pdm[pdm$experiment == 1, ]
pdm$N200_latencies <- pdm$N200_latencies/1000
pdm$RT <- pdm$RT/1000

head(pdm)
```

Regression in brms

Simple linear regression equations

- y_i are the observations of our dependent variable y for each observation i
- x_i are the observations of our independent variable

- $y_i \sim \mathrm{Normal}(\mu_i, \sigma^2)$
- $\mu_i = \beta_0 + \beta_1 x_i$

- You **may** have previously learned this, which is equivalent:
- $y_i = \beta_0 + \beta_1 x_i + \epsilon_i$
- $\epsilon_i \sim \mathrm{Normal}(0, \sigma^2)$

Simple linear regression using base R

```
# install.packages('brms')
library(brms)
```

- brms nicely follows the same standard R formula syntax that base R and many packages follow
- Here is an example of the base R code:

```
simple_lm <- lm(RT ~ N200_latencies, data=pdm)
summary(simple_lm)
```

Simple linear regression using brms

```r
# install.packages('brms')
library(brms)
```

- brms nicely follows the same standard R formula syntax that base R and many packages follow
- Here is an example of brms code:
- Note that we used the **default *Stan* prior**

```r
bayes_lm <- brm(RT ~ N200_latencies, data=pdm)
summary(bayes_lm)
```

What is the brms code to estimate a linear regression with *RT* as the dependent variable, *N200_latencies* and *N200_amplitudes* as the independent variables, and *an interaction term*?

Hint: Read the help file in RStudio using ?brm

Extended linear regression using brms

- Note there are at least three different methods that yield the same solution.
- Here is one **short** example solution:

```
bayes_lmext <- brm(RT ~ N200_latencies*N200_amplitudes,
    data=pdm)
summary(bayes_lmext)
```

Extended linear regression equations

- y_i are the observations of our dependent variable y for each observation i
- x_{ki} are the observations of our independent variables for each indepdent variable k

- $y_i \sim \mathrm{Normal}(\mu_i, \sigma^2)$
- $\mu_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{3i} + \ldots$

- You **may** have previously learned this, which is equivalent:
- $y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{3i} + \ldots + \epsilon_i$
- $\epsilon_i \sim \mathrm{Normal}(0, \sigma^2)$

ANOVA in brms

A simple 3-level, 1-way Bayesian “ANOVA”

- Let’s think about the signal-to-noise (SNR) *condition* effect on response time *RT*

```
bayes_anova <- brm(RT ~ factor(condition), data=pdm)
summary(bayes_anova)
```

A simple 2-way Bayesian "ANOVA"

- Let's think about both the signal-to-noise (SNR) *condition*, the effect of *accuracy*, and the interaction effect on response time (*RT*)

```r
bayes_anova <-
brm(RT ~ factor(condition)*factor(accuracy), data=pdm)
summary(bayes_anova)
```

Reading brms output

What effects are significant in the previous model?

What effects are "significant"?

- Use Bayesian probability as evidence for an effect
- For what range are you 95% certain of an effect
- What if this overlaps 0?

```
Population-Level Effects:
                                  Estimate Est.Error l-95% CI u-95% CI Rhat
Intercept                             0.84      0.01     0.82     0.86 1.00
factorcondition1                     -0.05      0.01    -0.07    -0.02 1.00
factorcondition2                     -0.02      0.01    -0.04     0.01 1.00
factoraccuracy1                      -0.03      0.01    -0.06    -0.01 1.00
factorcondition1:factoraccuracy1    -0.03      0.02    -0.06     0.00 1.00
factorcondition2:factoraccuracy1    -0.06      0.02    -0.09    -0.02 1.00
                                  Bulk_ESS Tail_ESS
Intercept                             2108     2570
factorcondition1                      2060     2427
factorcondition2                      1973     2421
factoraccuracy1                       1967     2563
factorcondition1:factoraccuracy1     1943     2339
factorcondition2:factoraccuracy1     1957     2221

Family Specific Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sigma     0.24      0.00     0.23     0.24 1.00     3513     2710
```

Convergence diagnostics: Rhat

- The *Gelman-Rubin convergence diagnostic* statistic `Rhat`
- `Rhat` give the ratio of the between-chain to within-chain variance.
- `Rhat` be close to 1

- **Rule of thumb:** ≤ 1.01 for regression models
- **Rule of thumb:** ≤ 1.10 for complex hierarchical models

Convergence diagnostics: ESS

- The effective sample size `ESS` should be large for appropriate posterior distribution estimates.
- `ESS` statistics penalize the true number of posterior samples by the Markov Chain autocorrelation
- `Bulk_ESS` is best `ESS` diagnostic the mean posterior `Estimate`
- `Tail_ESS` is best `ESS` diagnostic the 95% credible intervals `CI`

- **Rule of thumb:** > 100

Did the model converge?

```
Population-Level Effects:
                                 Estimate Est.Error l-95% CI u-95% CI Rhat
Intercept                            0.84      0.01     0.82     0.86 1.00
factorcondition1                    -0.05      0.01    -0.07    -0.02 1.00
factorcondition2                    -0.02      0.01    -0.04     0.01 1.00
factoraccuracy1                     -0.03      0.01    -0.06    -0.01 1.00
factorcondition1:factoraccuracy1    -0.03      0.02    -0.06     0.00 1.00
factorcondition2:factoraccuracy1    -0.06      0.02    -0.09    -0.02 1.00
                                 Bulk_ESS Tail_ESS
Intercept                            2108     2570
factorcondition1                     2060     2427
factorcondition2                     1973     2421
factoraccuracy1                      1967     2563
factorcondition1:factoraccuracy1     1943     2339
factorcondition2:factoraccuracy1     1957     2221

Family Specific Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sigma     0.24      0.00     0.23     0.24 1.00     3513     2710
```

Structured individual differences

Simpson’s paradox

- Simpson’s paradox refers to the fact that the the effects could be reversed on the participant level

Fitting a regression per participant

- One way to avoid Simpson's paradox is to fit regression models to each participant / individual
- However the sample size per model is greatly reduced.
- Also this method is less robust to contaminant data

Structured individual differences

- Another method is to assume that the participant-level differences are related
- Knowledge about other participants helps estimate parameters of new participants
- The sample size per participant necessary to estimate the model is much less than the individual model strategy
- Outlier participants are less influential

Hierarchical linear regression equations

- y_{ip} are the observations of our dependent variable y for each observation i and participant p
- x_{ip} are the observations of our independent variable
- β_{0p} are participant level intercepts
- β_{1p} be the observations of our independent variables
- Both β_{0p} and β_{1p} come from *hierarchical* distributions

- $y_i \sim \mathrm{Normal}(\mu_i,\ \sigma^2)$
- $\mu_i = \beta_{0p} + \beta_{1p} x_{ip}$
- $\beta_{0p} \sim \mathrm{Normal}(\mu_0,\ \sigma_0^2)$
- $\beta_{1p} \sim \mathrm{Normal}(\mu_1,\ \sigma_1^2)$

Multi-level regression in brms

Linear regression with random intercepts

```
# install.packages('brms')
library(brms)
```

- Here is an example of brms code for random intercepts:
- This may take a couple of minutes to run

```
bayes_randint <-
brm(RT ~ N200_latencies + (1|subject), data=pdm)
summary(bayes_randint)
```

- Press STOP in the top right of your RStudio console to end the model fitting

Linear regression with random slopes

```
# install.packages('brms')
library(brms)
```

- Here is an example of brms code for random slopes:
- DO NOT RUN THIS NOW
- This will take some time to run

```
bayes_randeffect <-
brm(RT ~ N200_latencies + (N200_latencies|subject),
data=pdm)
summary(bayes_randeffect)
```

What is the brms code to estimate a linear regression with *RT* as the dependent variable, *N200_latencies* and *N200_amplitudes* as the independent variables, *an interaction term*, and a random intercept for each *subject*?

Final hierarchical regression model in brms

- Note there are a few different methods that yield the same solution.
- Here is one **short** example solution:

```
bayes_final <-
brm(RT ~ (1|subject) + N200_latencies*N200_amplitudes,
data=pdm)
summary(bayes_final)
```

- Press STOP in the top right of your RStudio console to end the model fitting

Now let’s talk about cognitive modeling!