E1 memory

R. Cassano-Coleman

2025-09-30

This notebook analyzes memory using Bayesian binomial generalized linear mixed effects models (GLMMs).

Set up

```
set.seed(15000)
# other seeds for stability checks
#set.seed(20)
#set.seed(20000)
```

```
data <- read_csv('../data/E1-E2-E4/memory.csv')
```

```
## Rows: 3150 Columns: 6
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): response, scramble, Musician
## dbl (3): exp_subject_id, Trial_Nr, yrs_mus_exp
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Convert variables to factors.

```
data %<>%
  mutate(exp_subject_id = as.factor(exp_subject_id),
         response = ifelse(response == 'Correct', TRUE, FALSE),
         scramble = factor(scramble, levels = c('8B', '2B', '1B')),
         Musician = factor(Musician, levels = c('No', 'Yes'))) %>%
  filter(!is.na(response))
```

Set the contrast for condition.

```
contrasts(data$scramble) <- contr.treatment(3)
print(contrasts(data$scramble))
```

```
##    2 3
## 8B 0 0
## 2B 1 0
## 1B 0 1
```

Set the musician/non-musician contrast.

```
contrasts(data$Musician) <- contr.sum(2)
print(contrasts(data$Musician))
```

```
##     [,1]
```

```
## No     1
## Yes   -1
```

Main analysis

Priors

Priors are expressed in log(odds) space.

Intercept: Given that chance is 50%, we assume that participants will perform somewhere between chance and ceiling. We expect the center of the distribution of accuracy to be somewhere around 75% or 80%. If we use a center of 80% and an SD of 1, 95% of the values fall between 35.1% and 96.7%.

```
prior_intercept <- set_prior('normal(log(0.8 / (1 - 0.8)), 1)', class = 'Intercept')
```

Group: We might expect musicians to do slightly better than non-musicians, on average.

In this range, a difference in 0.25 log odds gives us about a 5% decrease in accuracy.

```
prior_mus <- set_prior('normal(-0.25, 1)', coef = 'Musician1')
```

Scramble: We expect performance to improve as scramble level decreases. If we code 8B as reference level, then we expect 8B > 2B and 8B > 1B.

Since we're keeping the musician slope at SD = 1, we'll keep these (and the interactions) at SD = 1. This seems to be a pretty weak prior.

```
prior_scramble2B <- set_prior('normal(-0.1, 1)', coef = 'scramble2')
prior_scramble1B <- set_prior('normal(-0.2, 1)', coef = 'scramble3')
```

Interaction: We expect no interaction between group and scramble.

```
prior_int2B <- set_prior('normal(0, 1)', coef = 'Musician1:scramble2')
prior_int1B <- set_prior('normal(0, 1)', coef = 'Musician1:scramble3')
```

Random slope for subjects: *Leave this as default for now, may update.*

Main model with group and condition

```
get_prior(response ~ Musician + scramble + (1 | exp_subject_id), data = data)
```

```
##                 prior     class      coef          group resp dpar nlpar lb ub
##                (flat)         b                                              
##                (flat)         b Musician1                                    
##                (flat)         b scramble2                                    
##                (flat)         b scramble3                                    
##  student_t(3, 0, 2.5) Intercept                                              
##  student_t(3, 0, 2.5)        sd                                         0    
##  student_t(3, 0, 2.5)        sd           exp_subject_id                0    
##  student_t(3, 0, 2.5)        sd Intercept exp_subject_id                0    
##  student_t(3, 0, 2.5)     sigma                                         0    
##        source
##       default
##  (vectorized)
##  (vectorized)
##  (vectorized)
##       default
##       default
##  (vectorized)
##  (vectorized)
##       default
```

```
mus_scram <- brm(response ~ Musician + scramble + (1 | exp_subject_id), data = data,
                 family = bernoulli(),
                 prior = c(prior_intercept, prior_mus,
                           prior_scramble2B, prior_scramble1B),
                 save_pars = save_pars(all = TRUE), iter = 20000, silent = 1,
                 file = 'models/E1_mus_scram')
```

```
## Compiling Stan program...
```

```
## Trying to compile a simple C file
```

```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## using C compiler: ‘Apple clang version 16.0.0 (clang-1600.0.26.6)’
## using SDK: ‘MacOSX15.2.sdk’
## clang -arch arm64 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Framew
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/StanHeade
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/include/Eigen/src/Core
##   679 | #include <cmath>
##       |          ^~~~~~
## 1 error generated.
## make: *** [foo.o] Error 1
```

```
## Start sampling
```

```
## 
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.000191 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 1.91 seconds.
```

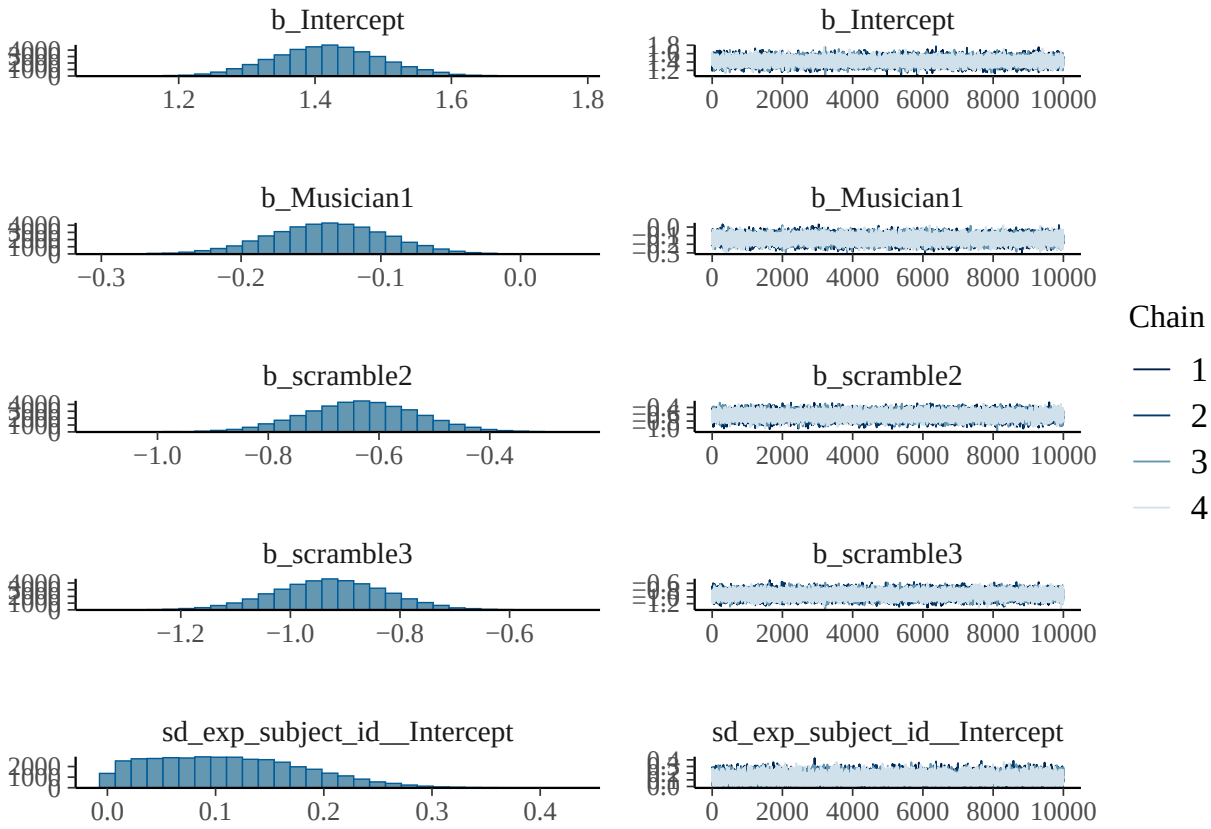
```
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 1: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 1: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 1: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 1: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 1: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 1: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 1: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 1: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 1: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 1: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 1: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 19.143 seconds (Warm-up)
## Chain 1:                16.625 seconds (Sampling)
## Chain 1:                35.768 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0.000111 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 1.11 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 2: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 2: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 2: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 2: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 2: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 2: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 2: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 2: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 2: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 2: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 2: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 18.439 seconds (Warm-up)
## Chain 2:                16.897 seconds (Sampling)
## Chain 2:                35.336 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0.000112 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 1.12 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:     1 / 20000 [  0%]  (Warmup)
```

```
## Chain 3: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 3: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 3: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 3: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 3: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 3: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 3: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 3: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 3: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 3: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 3: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 18.59 seconds (Warm-up)
## Chain 3:                16.494 seconds (Sampling)
## Chain 3:                35.084 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0.000108 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 1.08 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 4: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 4: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 4: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 4: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 4: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 4: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 4: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 4: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 4: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 4: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 4: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 19.013 seconds (Warm-up)
## Chain 4:                16.232 seconds (Sampling)
## Chain 4:                35.245 seconds (Total)
## Chain 4:
```

```
## Warning: There were 7 divergent transitions after warmup. See
## https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

```
plot(mus_scram)
```




```
print(summary(mus_scram), digits = 4)
```

```
## Warning: There were 7 divergent transitions after warmup. Increasing
## adapt_delta above 0.8 may help. See
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: response ~ Musician + scramble + (1 | exp_subject_id)
##    Data: data (Number of observations: 3094)
##   Draws: 4 chains, each with iter = 20000; warmup = 10000; thin = 1;
##          total post-warmup draws = 40000
##
## Multilevel Hyperparameters:
## ~exp_subject_id (Number of levels: 102)
##               Estimate Est.Error l-95% CI u-95% CI   Rhat Bulk_ESS Tail_ESS
## sd(Intercept)   0.1115    0.0693   0.0055   0.2569 1.0007     9061    13973
##
## Regression Coefficients:
##           Estimate Est.Error l-95% CI u-95% CI   Rhat Bulk_ESS Tail_ESS
## Intercept   1.4181    0.0794   1.2655   1.5750 1.0002    31306    26282
## Musician1  -0.1375    0.0419  -0.2206  -0.0565 1.0003    45374    27324
## scramble2  -0.6319    0.1033  -0.8350  -0.4311 1.0001    37188    31040
## scramble3  -0.9272    0.1015  -1.1272  -0.7309 1.0000    35912    29864
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```r
emm_mus_scram_s <- emmeans(mus_scram, specs = "scramble")
summary(emm_mus_scram_s)
```

```
##  scramble emmean lower.HPD upper.HPD
##  8B        1.418     1.267     1.576
##  2B        0.786     0.651     0.920
##  1B        0.491     0.364     0.621
##
## Results are averaged over the levels of: Musician
## Point estimate displayed: median
## Results are given on the logit (not the response) scale.
## HPD interval probability: 0.95
```

```r
contrast(emm_mus_scram_s, method = "pairwise")
```

```
##  contrast estimate lower.HPD upper.HPD
##  8B - 2B     0.632     0.435     0.838
##  8B - 1B     0.927     0.725     1.121
##  2B - 1B     0.295     0.115     0.482
##
## Results are averaged over the levels of: Musician
## Point estimate displayed: median
## Results are given on the log odds ratio (not the response) scale.
## HPD interval probability: 0.95
```

```r
emm_mus_scram_ms <- emmeans(mus_scram, specs = c("Musician", "scramble"))
summary(emm_mus_scram_ms)
```

```
##  Musician scramble emmean lower.HPD upper.HPD
##  No       8B        1.280     1.108     1.457
##  Yes      8B        1.555     1.375     1.728
##  No       2B        0.649     0.492     0.805
##  Yes      2B        0.923     0.767     1.085
##  No       1B        0.353     0.204     0.507
##  Yes      1B        0.628     0.472     0.778
##
## Point estimate displayed: median
## Results are given on the logit (not the response) scale.
## HPD interval probability: 0.95
```

```r
contrast(emm_mus_scram_ms, method = "pairwise")
```

```
##  contrast        estimate lower.HPD upper.HPD
##  No 8B - Yes 8B   -0.2747   -0.4396    -0.112
##  No 8B - No 2B     0.6315    0.4346     0.838
##  No 8B - Yes 2B    0.3562    0.0907     0.611
##  No 8B - No 1B     0.9273    0.7252     1.121
##  No 8B - Yes 1B    0.6523    0.3911     0.907
##  Yes 8B - No 2B    0.9062    0.6508     1.172
##  Yes 8B - Yes 2B   0.6315    0.4346     0.838
##  Yes 8B - No 1B    1.2023    0.9371     1.457
##  Yes 8B - Yes 1B   0.9273    0.7252     1.121
##  No 2B - Yes 2B   -0.2747   -0.4396    -0.112
##  No 2B - No 1B     0.2950    0.1146     0.482
##  No 2B - Yes 1B    0.0204   -0.2211     0.268
##  Yes 2B - No 1B    0.5697    0.3245     0.814
```

```
##  Yes 2B - Yes 1B   0.2950    0.1146     0.482
##  No 1B - Yes 1B   -0.2747   -0.4396    -0.112
##
## Point estimate displayed: median
## Results are given on the log odds ratio (not the response) scale.
## HPD interval probability: 0.95
```

Main effects

```
main_BF <- describe_posterior(mus_scram,
                              estimate = "median", dispersion = TRUE,
                              ci = .95, ci_method = "HDI",
                              test = c("bayes_factor"))
print(main_BF, digits = 4)
```

```
## Summary of Posterior Distribution
##
## Parameter   |  Median |    MAD |         95% CI |       BF |  Rhat |        ESS
## ------------------------------------------------------------------------------
## (Intercept) |  1.4178 | 0.0787 | [ 1.27,  1.58] | 2.25e+18 | 1.000 | 31216.0000
## Musician1   | -0.1373 | 0.0420 | [-0.22, -0.06] |     9.75 | 1.000 | 45120.0000
## scramble2   | -0.6315 | 0.1030 | [-0.84, -0.43] | 2.85e+05 | 1.000 | 37246.0000
## scramble3   | -0.9273 | 0.1014 | [-1.12, -0.73] | 8.52e+07 | 1.000 | 35845.0000
```

Moderate evidence for a main effect of group.

To get the main effect of scramble level, fit the "null" model with group only to compare.

```
mus_only <- brm(response ~ Musician + (1 | exp_subject_id), data = data,
                family = bernoulli(),
                prior = c(prior_intercept, prior_mus),
                save_pars = save_pars(all = TRUE), iter = 20000, silent = 1,
                file = 'models/E1_mus_only')
```

```
## Compiling Stan program...
```

```
## Trying to compile a simple C file
```

```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## using C compiler: 'Apple clang version 16.0.0 (clang-1600.0.26.6)'
## using SDK: 'MacOSX15.2.sdk'
## clang -arch arm64 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Framew
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/StanHeade
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/include/Eigen/src/Core
##   679 | #include <cmath>
##       |          ^~~~~~
## 1 error generated.
## make: *** [foo.o] Error 1
```

```
## Start sampling
```

```
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.000186 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 1.86 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 1: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 1: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 1: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 1: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 1: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 1: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 1: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 1: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 1: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 1: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 1: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 17.209 seconds (Warm-up)
## Chain 1:                20.102 seconds (Sampling)
## Chain 1:                37.311 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0.000101 seconds
```

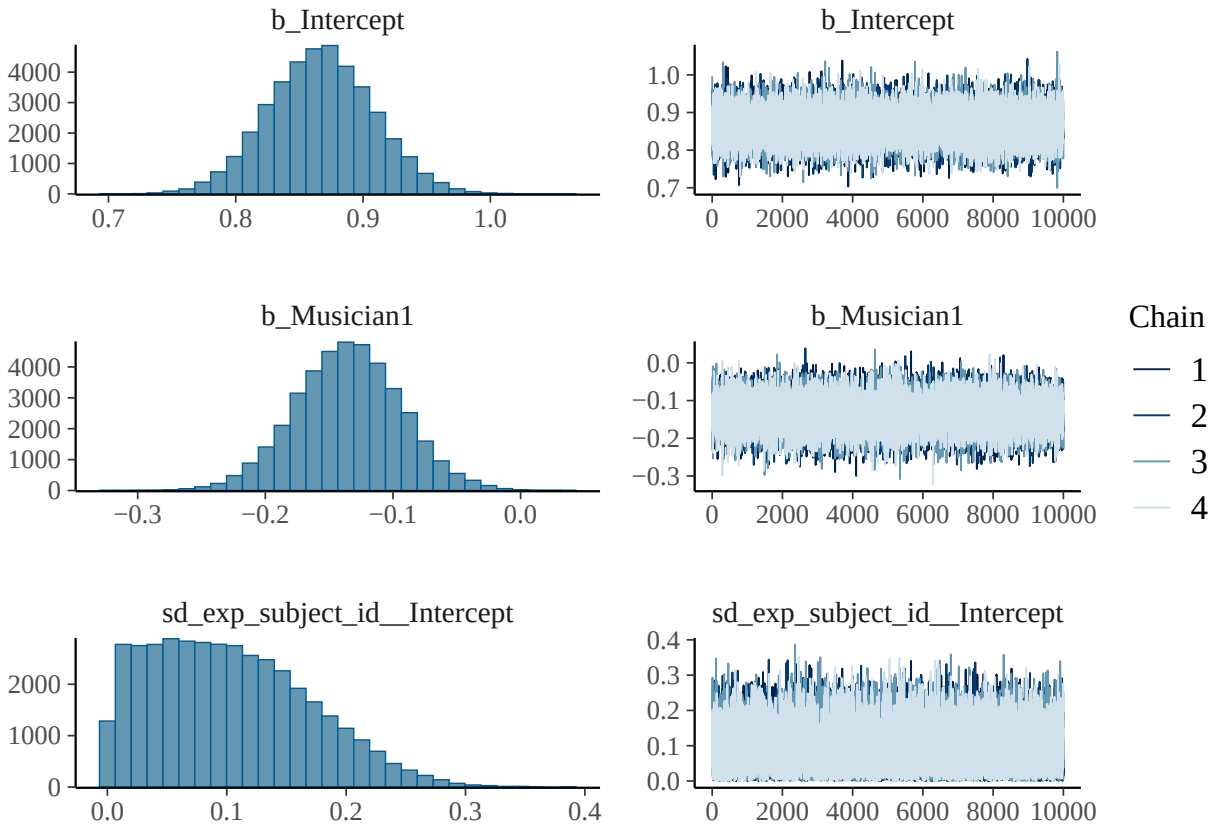
```
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 1.01 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 2: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 2: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 2: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 2: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 2: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 2: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 2: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 2: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 2: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 2: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 2: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 17.13 seconds (Warm-up)
## Chain 2:                15.275 seconds (Sampling)
## Chain 2:                32.405 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0.000105 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 1.05 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 3: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 3: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 3: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 3: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 3: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 3: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 3: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 3: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 3: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 3: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 3: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 17.134 seconds (Warm-up)
## Chain 3:                15.267 seconds (Sampling)
## Chain 3:                32.401 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0.000102 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 1.02 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
```

```
## Chain 4: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 4: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 4: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 4: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 4: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 4: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 4: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 4: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 4: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 4: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 4: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 4: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 17.695 seconds (Warm-up)
## Chain 4:                15.28 seconds (Sampling)
## Chain 4:                32.975 seconds (Total)
## Chain 4:
```

```
## Warning: There were 1 divergent transitions after warmup. See
## https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

```
plot(mus_only)
```




```
print(summary(mus_only), digits = 4)
```

```
## Warning: There were 1 divergent transitions after warmup. Increasing
## adapt_delta above 0.8 may help. See
```

```
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: response ~ Musician + (1 | exp_subject_id)
##    Data: data (Number of observations: 3094)
##   Draws: 4 chains, each with iter = 20000; warmup = 10000; thin = 1;
##          total post-warmup draws = 40000
##
## Multilevel Hyperparameters:
## ~exp_subject_id (Number of levels: 102)
##               Estimate Est.Error l-95% CI u-95% CI   Rhat Bulk_ESS Tail_ESS
## sd(Intercept)   0.1022    0.0656   0.0052   0.2422 1.0001    10885    18340
##
## Regression Coefficients:
##           Estimate Est.Error l-95% CI u-95% CI   Rhat Bulk_ESS Tail_ESS
## Intercept   0.8668    0.0413   0.7864   0.9478 1.0004    44894    27007
## Musician1  -0.1351    0.0414  -0.2167  -0.0534 1.0002    44459    26997
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
BF_scramble <- bayes_factor(mus_scram, mus_only)
```

```
## Iteration: 1
## Iteration: 2
## Iteration: 3
## Iteration: 4
## Iteration: 5
## Iteration: 1
## Iteration: 2
## Iteration: 3
## Iteration: 4
## Iteration: 5
```

```
print(BF_scramble)
```

```
## Estimated Bayes factor in favor of mus_scram over mus_only: 236485054159734656.00000
```

Very strong evidence for a main effect of scramble condition.

Interaction between group and condition?

Add an interaction between group and condition, and compare the model with the interaction to the one without.

```
mus_scram_int <- brm(response ~ Musician*scramble + (1 | exp_subject_id), data = data,
                 family = bernoulli(),
                 prior = c(prior_intercept, prior_mus,
                           prior_scramble2B, prior_scramble1B,
                           prior_int2B, prior_int1B),
                 save_pars = save_pars(all = TRUE), iter = 20000, silent = 1,
                 file = 'models/E1_mus_scram_int')
```

```
## Compiling Stan program...
```

```
## Trying to compile a simple C file
```

```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## using C compiler: 'Apple clang version 16.0.0 (clang-1600.0.26.6)'
## using SDK: 'MacOSX15.2.sdk'
## clang -arch arm64 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Framew
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/StanHeade
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/include/Eigen/src/Core
##   679 | #include <cmath>
##       |          ^~~~~~
## 1 error generated.
## make: *** [foo.o] Error 1
```

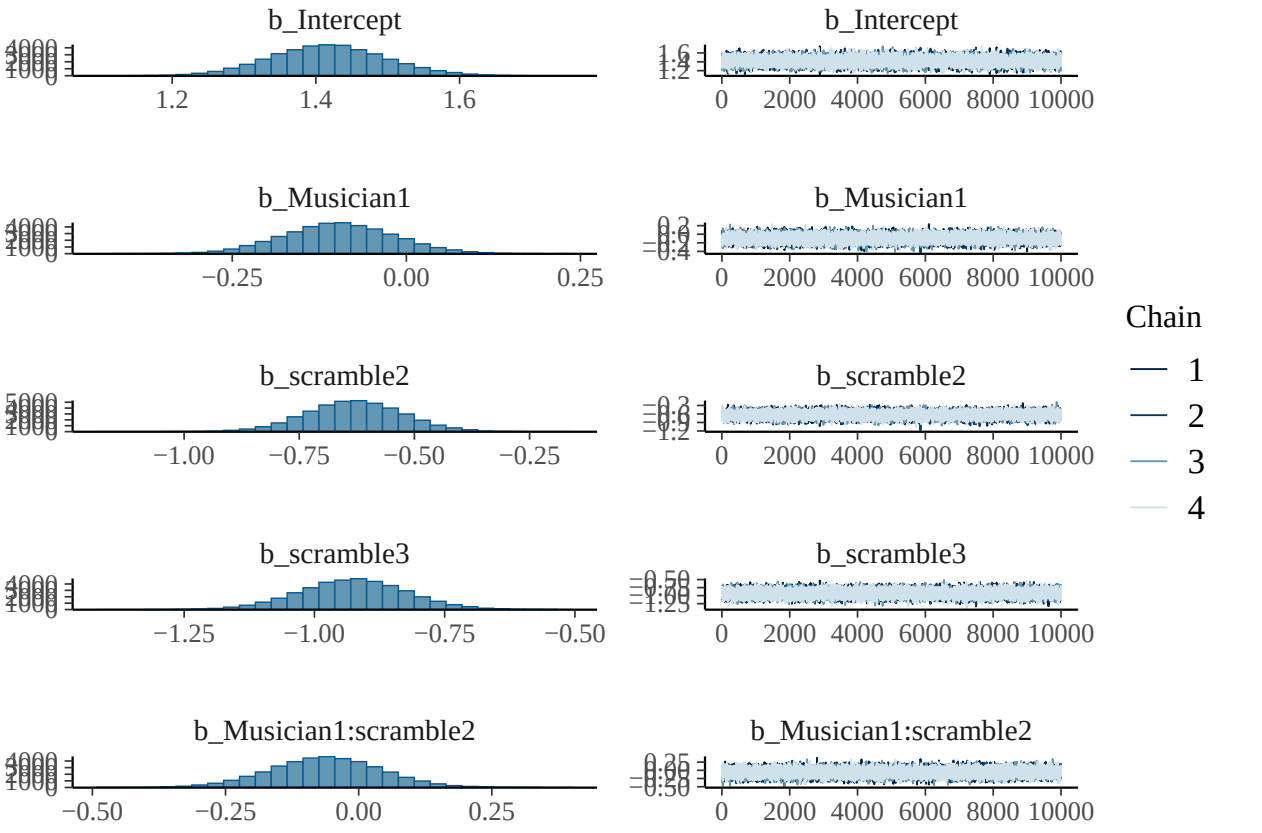
```
## Start sampling
```

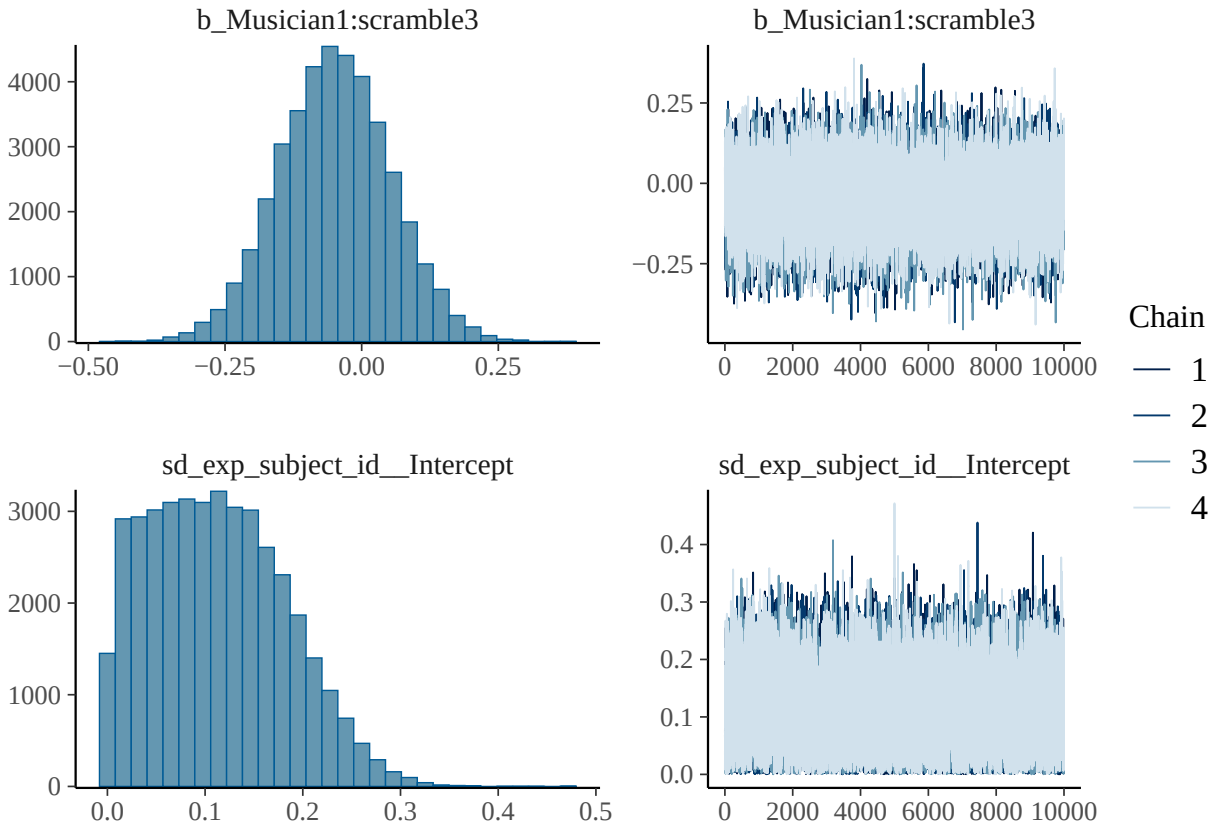
```
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.000212 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 2.12 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 1: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 1: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 1: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 1: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 1: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 1: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 1: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 1: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 1: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 1: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 1: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 18.649 seconds (Warm-up)
## Chain 1:                16.293 seconds (Sampling)
## Chain 1:                34.942 seconds (Total)
```

```
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0.000108 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 1.08 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 2: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 2: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 2: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 2: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 2: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 2: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 2: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 2: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 2: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 2: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 2: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 18.677 seconds (Warm-up)
## Chain 2:                16.301 seconds (Sampling)
## Chain 2:                34.978 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0.000104 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 1.04 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 3: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 3: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 3: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 3: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 3: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 3: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 3: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 3: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 3: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 3: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 3: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 18.764 seconds (Warm-up)
## Chain 3:                18.897 seconds (Sampling)
## Chain 3:                37.661 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).
## Chain 4:
```

```
## Chain 4: Gradient evaluation took 0.000107 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 1.07 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 4: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 4: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 4: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 4: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 4: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 4: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 4: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 4: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 4: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 4: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 4: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 18.532 seconds (Warm-up)
## Chain 4:                16.3 seconds (Sampling)
## Chain 4:                34.832 seconds (Total)
## Chain 4:
```

```
plot(mus_scram_int)
```






```
print(summary(mus_scram_int), digits = 4)
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: response ~ Musician * scramble + (1 | exp_subject_id)
##    Data: data (Number of observations: 3094)
##   Draws: 4 chains, each with iter = 20000; warmup = 10000; thin = 1;
##          total post-warmup draws = 40000
##
## Multilevel Hyperparameters:
## ~exp_subject_id (Number of levels: 102)
##               Estimate Est.Error l-95% CI u-95% CI   Rhat Bulk_ESS Tail_ESS
## sd(Intercept)   0.1120    0.0692   0.0057   0.2547 1.0001     9054    14441
##
## Regression Coefficients:
##                     Estimate Est.Error l-95% CI u-95% CI   Rhat Bulk_ESS
## Intercept             1.4175    0.0802   1.2626   1.5780 1.0002    29745
## Musician1            -0.0969    0.0798  -0.2532   0.0606 1.0002    19404
## scramble2            -0.6303    0.1035  -0.8343  -0.4273 1.0001    34721
## scramble3            -0.9258    0.1021  -1.1273  -0.7242 1.0000    34882
## Musician1:scramble2  -0.0624    0.1036  -0.2672   0.1399 1.0002    22582
## Musician1:scramble3  -0.0494    0.1019  -0.2486   0.1514 1.0002    22488
##                     Tail_ESS
## Intercept              26997
## Musician1              23625
## scramble2              28980
## scramble3              29301
## Musician1:scramble2    27591
```

```
## Musician1:scramble3    26722
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
BF_int <- bayes_factor(mus_scram_int, mus_scram)
```

```
## Iteration: 1
## Iteration: 2
## Iteration: 3
## Iteration: 4
## Iteration: 5
## Iteration: 1
## Iteration: 2
## Iteration: 3
## Iteration: 4
## Iteration: 5
```

```
print(BF_int)
```

```
## Estimated Bayes factor in favor of mus_scram_int over mus_scram: 0.01042
```

Strong evidence against an interaction between group and condition.

Figure 2A

Create a helper function for the conversion from log odds to probability.

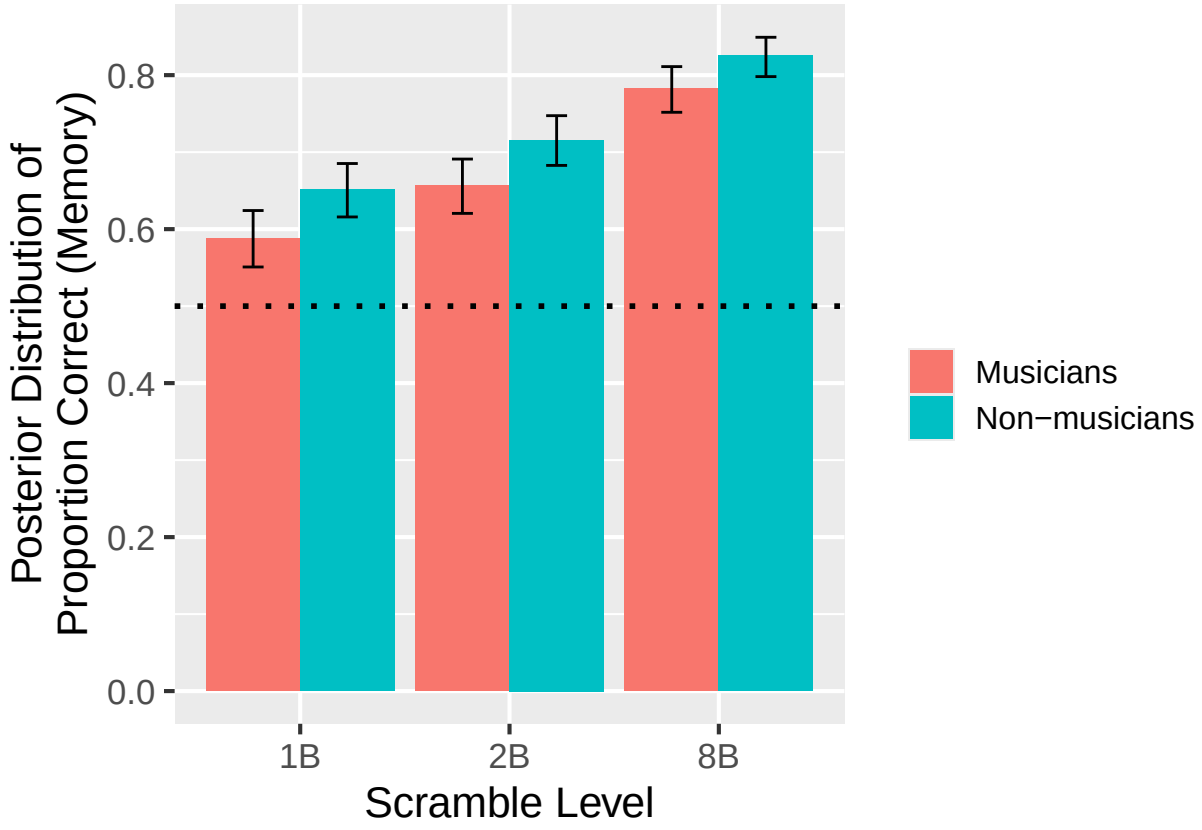
```
calculate_prob_from_logodds <- function(logodds) {
  return(exp(logodds) / (1 + exp(logodds)))
}
```

Visualize with posterior estimates and 95% CrI on the scale of accuracy.

```
posterior_est <- as.data.frame(emm_mus_scram_ms)
```

```
ggplot() +
  geom_col(aes(x = scramble, y = calculate_prob_from_logodds(emmean), fill = Musician),
           data = posterior_est,
           position = "dodge") +
  geom_errorbar(aes(x = scramble,
                    ymin = calculate_prob_from_logodds(lower.HPD),
                    ymax = calculate_prob_from_logodds(upper.HPD),
                    fill = Musician),
                data = posterior_est, position = position_dodge(width = 0.9), width = 0.2) +
  geom_hline(yintercept = 0.5, linetype = "dotted", color = "black", linewidth = 1) +
  theme_gray(base_size = 16) +
  scale_x_discrete(limits = rev) +
  #ylim(0, 0.85) +
  xlab('Scramble Level') +
  ylab('Posterior Distribution of\nProportion Correct (Memory)') +
  scale_fill_discrete(name="", labels=c('Musicians', 'Non-musicians')) +
  theme(legend.text = element_text(size = 12))
```

```
## Warning in geom_errorbar(aes(x = scramble, ymin =
## calculate_prob_from_logodds(lower.HPD), : Ignoring unknown aesthetics: fill
```




```
ggsave('../figures/Fig2A_memory.png', width = 7, height = 5)
```

Years of experience

Keep only the subjects for which we have years of experience data and average accuracy per condition.

```
yrs_exp <- data %>%
  filter(!is.na(yrs_mus_exp)) %>%
  group_by(exp_subject_id, scramble, yrs_mus_exp) %>%
  summarize(count = n(),
            n_correct = sum(response),
            accuracy = n_correct / count)
```

```
## `summarise()` has grouped output by 'exp_subject_id', 'scramble'. You can
## override using the `.groups` argument.
```

Priors

For this analysis, we're operating on the scale of accuracy. Because we don't see ceiling effects (i.e. participants aren't getting too close to perfect accuracy), a linear model is appropriate enough.

```
these_priors <- c(
  set_prior('normal(0.75, 0.1)', class = 'Intercept'),
  set_prior('normal(-0.1, 0.1)', coef = 'scramble2'),
  set_prior('normal(-0.2, 0.1)', coef = 'scramble3'),
  set_prior('normal(0, 0.1)', coef = 'yrs_mus_exp')
)
```

Main model

```
years_mus_scram <- brm(accuracy ~ scramble + yrs_mus_exp + (1|exp_subject_id), data = yrs_exp,
                  prior = these_priors,
                  save_pars = save_pars(all = TRUE), iter = 20000, silent = 1,
                  file = 'models/E1_years')
```

```
## Compiling Stan program...
```

```
## Trying to compile a simple C file
```

```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## using C compiler: 'Apple clang version 16.0.0 (clang-1600.0.26.6)'
## using SDK: 'MacOSX15.2.sdk'
## clang -arch arm64 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Framew
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/StanHeade
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/include/Eigen/src/Core
##   679 | #include <cmath>
##       |          ^~~~~~
## 1 error generated.
## make: *** [foo.o] Error 1
```

```
## Start sampling
```

```
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 5.3e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.53 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 1: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 1: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 1: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 1: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 1: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 1: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 1: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 1: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 1: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 1: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 1: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 1.483 seconds (Warm-up)
## Chain 1:                1.035 seconds (Sampling)
## Chain 1:                2.518 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 7e-06 seconds
```

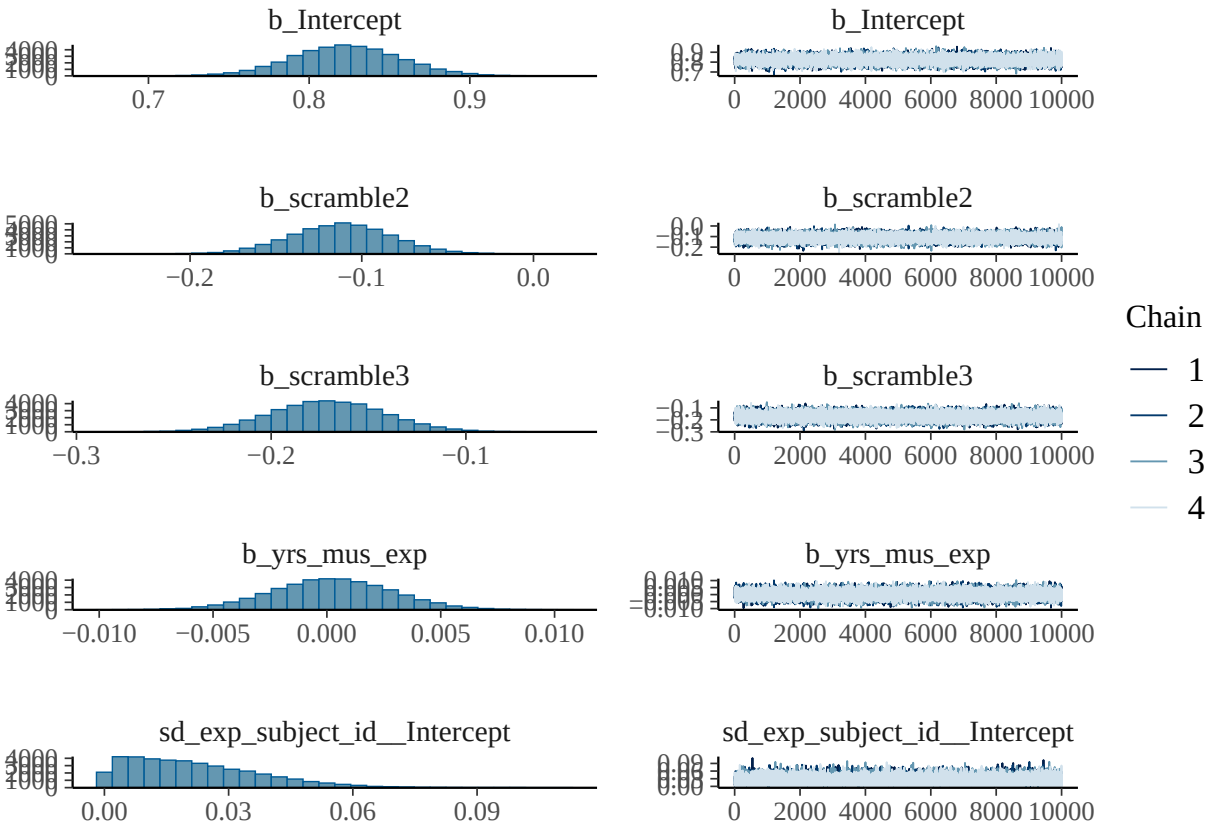
```
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.07 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 2: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 2: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 2: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 2: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 2: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 2: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 2: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 2: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 2: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 2: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 2: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 1.456 seconds (Warm-up)
## Chain 2:                0.947 seconds (Sampling)
## Chain 2:                2.403 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 8e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.08 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 3: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 3: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 3: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 3: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 3: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 3: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 3: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 3: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 3: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 3: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 3: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 1.441 seconds (Warm-up)
## Chain 3:                1.035 seconds (Sampling)
## Chain 3:                2.476 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 8e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.08 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
```

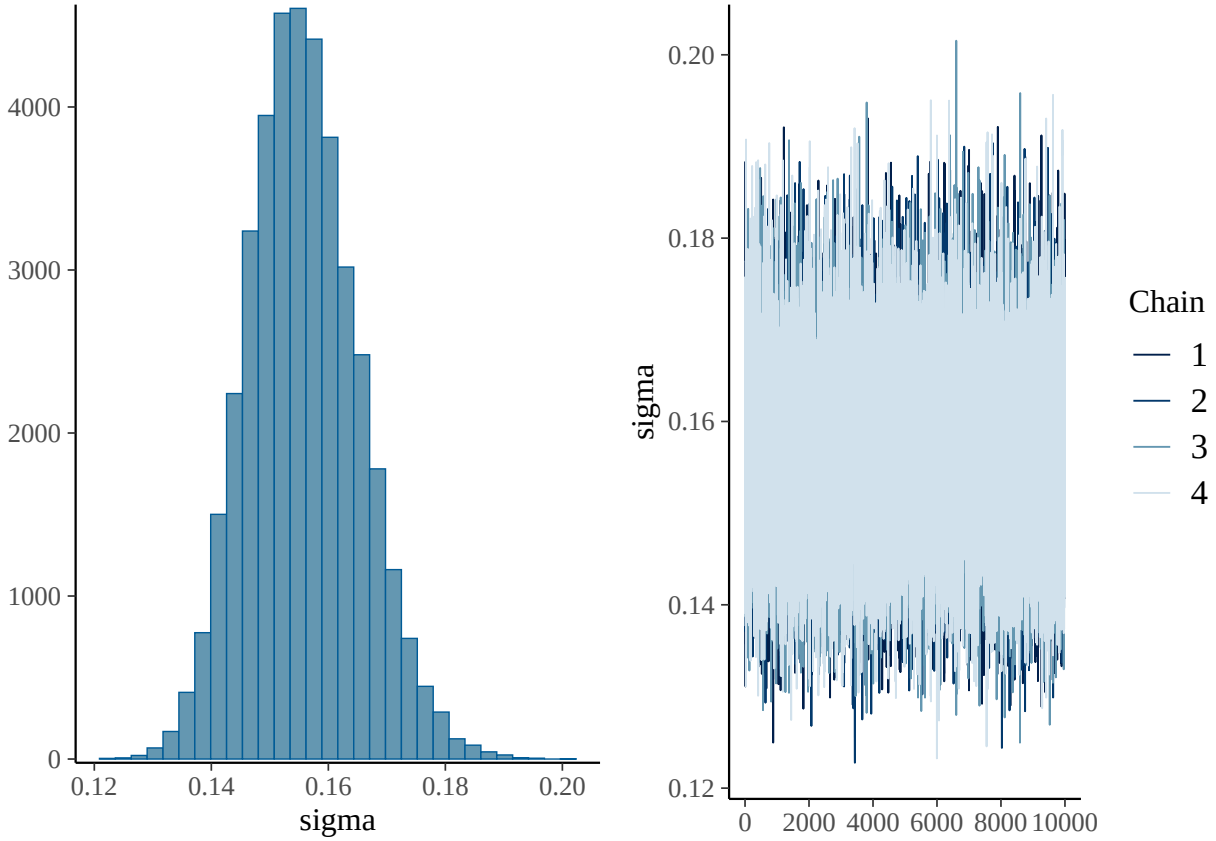
```
## Chain 4: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 4: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 4: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 4: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 4: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 4: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 4: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 4: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 4: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 4: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 4: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 4: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 1.467 seconds (Warm-up)
## Chain 4:                1.033 seconds (Sampling)
## Chain 4:                2.5 seconds (Total)
## Chain 4:
```

```
## Warning: There were 11 divergent transitions after warmup. See
## https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

```
plot(years_mus_scram)
```






```
print(summary(years_mus_scram), digits = 4)
```

```
## Warning: There were 11 divergent transitions after warmup. Increasing
## adapt_delta above 0.8 may help. See
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: accuracy ~ scramble + yrs_mus_exp + (1 | exp_subject_id)
##    Data: yrs_exp (Number of observations: 153)
##   Draws: 4 chains, each with iter = 20000; warmup = 10000; thin = 1;
##          total post-warmup draws = 40000
##
## Multilevel Hyperparameters:
## ~exp_subject_id (Number of levels: 51)
##               Estimate Est.Error l-95% CI u-95% CI   Rhat Bulk_ESS Tail_ESS
## sd(Intercept)   0.0218    0.0156   0.0009   0.0575 1.0003    15515    19236
##
## Regression Coefficients:
##             Estimate Est.Error l-95% CI u-95% CI   Rhat Bulk_ESS Tail_ESS
## Intercept     0.8230    0.0336   0.7572   0.8890 1.0001    46904    28752
## scramble2    -0.1120    0.0293  -0.1697  -0.0546 1.0001    45786    30468
## scramble3    -0.1709    0.0291  -0.2282  -0.1133 1.0001    46021    31258
## yrs_mus_exp   0.0002    0.0026  -0.0050   0.0054 1.0001    51418    27721
##
## Further Distributional Parameters:
##       Estimate Est.Error l-95% CI u-95% CI   Rhat Bulk_ESS Tail_ESS
## sigma   0.1557    0.0094   0.1384   0.1753 1.0001    45717    28129
```

```
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Null model (for plotting purposes)

```
years_mus <- brm(accuracy ~ yrs_mus_exp + (1|exp_subject_id), data = yrs_exp,
                 prior = c(
                   set_prior('normal(0.75, 0.1)', class = 'Intercept'),
                   set_prior('normal(0, 0.1)', coef = 'yrs_mus_exp')),
                 save_pars = save_pars(all = TRUE), iter = 20000, silent = 1,
                 file = 'models/E1_years_null')
```

```
## Compiling Stan program...
```

```
## Trying to compile a simple C file
```

```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## using C compiler: 'Apple clang version 16.0.0 (clang-1600.0.26.6)'
## using SDK: 'MacOSX15.2.sdk'
## clang -arch arm64 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Framew
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/StanHeade
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/include/Eigen/src/Core
##   679 | #include <cmath>
##       |          ^~~~~~~
## 1 error generated.
## make: *** [foo.o] Error 1
```

```
## Start sampling
```

```
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 5.2e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.52 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 1: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 1: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 1: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 1: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 1: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 1: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 1: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 1: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 1: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 1: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 1: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 1.364 seconds (Warm-up)
```

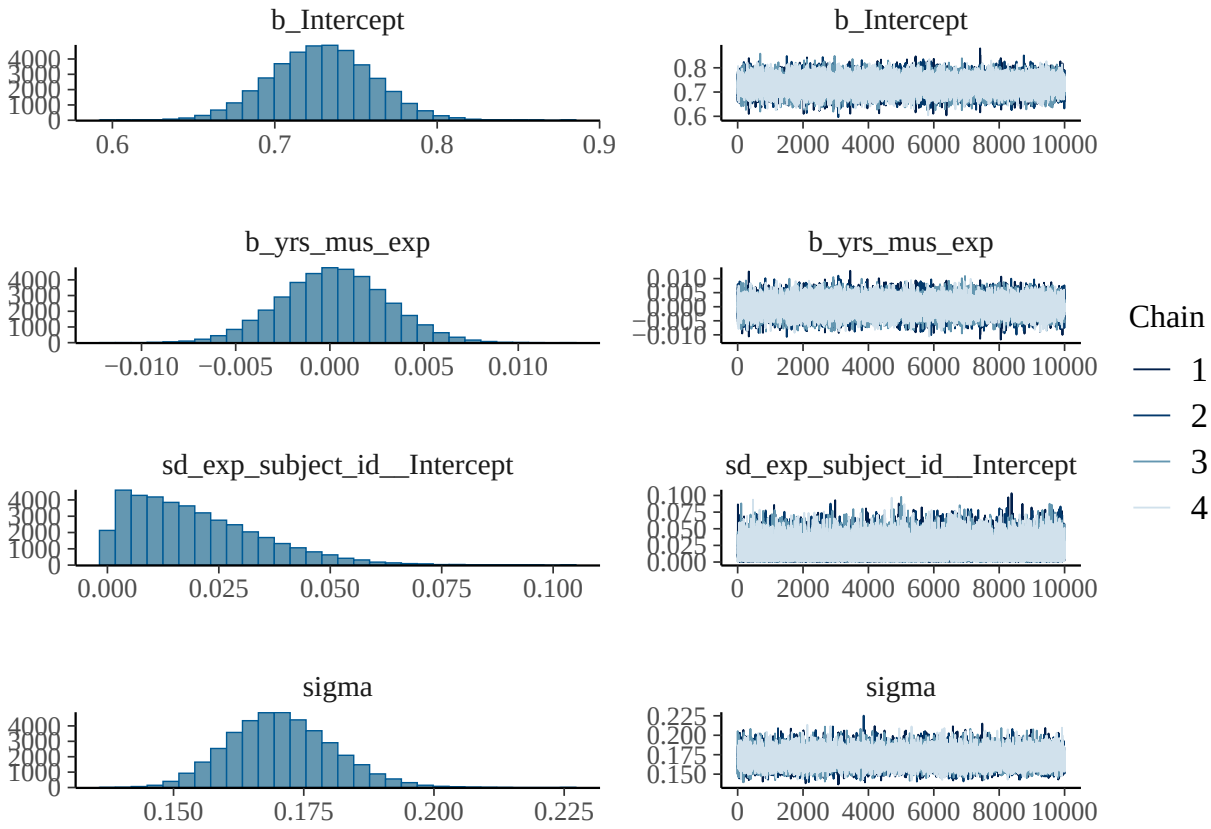
```
## Chain 1:                1.032 seconds (Sampling)
## Chain 1:                2.396 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 7e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.07 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 2: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 2: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 2: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 2: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 2: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 2: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 2: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 2: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 2: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 2: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 2: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 1.437 seconds (Warm-up)
## Chain 2:                0.986 seconds (Sampling)
## Chain 2:                2.423 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 7e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.07 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 3: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 3: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 3: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 3: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 3: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 3: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 3: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 3: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 3: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 3: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 3: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 1.29 seconds (Warm-up)
## Chain 3:                1.026 seconds (Sampling)
## Chain 3:                2.316 seconds (Total)
## Chain 3:
##
```

```
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 7e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.07 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:     1 / 20000 [  0%]  (Warmup)
## Chain 4: Iteration:  2000 / 20000 [ 10%]  (Warmup)
## Chain 4: Iteration:  4000 / 20000 [ 20%]  (Warmup)
## Chain 4: Iteration:  6000 / 20000 [ 30%]  (Warmup)
## Chain 4: Iteration:  8000 / 20000 [ 40%]  (Warmup)
## Chain 4: Iteration: 10000 / 20000 [ 50%]  (Warmup)
## Chain 4: Iteration: 10001 / 20000 [ 50%]  (Sampling)
## Chain 4: Iteration: 12000 / 20000 [ 60%]  (Sampling)
## Chain 4: Iteration: 14000 / 20000 [ 70%]  (Sampling)
## Chain 4: Iteration: 16000 / 20000 [ 80%]  (Sampling)
## Chain 4: Iteration: 18000 / 20000 [ 90%]  (Sampling)
## Chain 4: Iteration: 20000 / 20000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 1.379 seconds (Warm-up)
## Chain 4:                1.028 seconds (Sampling)
## Chain 4:                2.407 seconds (Total)
## Chain 4:
```

```
## Warning: There were 1 divergent transitions after warmup. See
## https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

```
plot(years_mus)
```




```
print(summary(years_mus), digits = 4)
```

```
## Warning: There were 1 divergent transitions after warmup. Increasing
## adapt_delta above 0.8 may help. See
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: accuracy ~ yrs_mus_exp + (1 | exp_subject_id)
##    Data: yrs_exp (Number of observations: 153)
##   Draws: 4 chains, each with iter = 20000; warmup = 10000; thin = 1;
##          total post-warmup draws = 40000
##
## Multilevel Hyperparameters:
## ~exp_subject_id (Number of levels: 51)
##               Estimate Est.Error l-95% CI u-95% CI   Rhat Bulk_ESS Tail_ESS
## sd(Intercept)   0.0198    0.0147   0.0008   0.0544 1.0005    22476    22541
##
## Regression Coefficients:
##             Estimate Est.Error l-95% CI u-95% CI   Rhat Bulk_ESS Tail_ESS
## Intercept     0.7288    0.0313   0.6674   0.7900 1.0001    66575    28625
## yrs_mus_exp   0.0002    0.0028  -0.0053   0.0058 1.0003    69045    29948
##
## Further Distributional Parameters:
##       Estimate Est.Error l-95% CI u-95% CI   Rhat Bulk_ESS Tail_ESS
## sigma   0.1707    0.0100   0.1525   0.1915 1.0002    68939    28977
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
```

```
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
yrs_BF <- describe_posterior(years_mus_scram,
                             estimate = "median", dispersion = TRUE,
                             ci = .95, ci_method = "HDI",
                             test = c("bayes_factor"))
print(yrs_BF, digits = 4)
```

```
## Summary of Posterior Distribution
##
## Parameter   |  Median |    MAD |         95% CI |       BF |  Rhat |        ESS
## ------------------------------------------------------------------------------
## (Intercept) |  0.8228 | 0.0334 | [ 0.76,  0.89] | 2.66e+29 | 1.000 | 46686.0000
## scramble2   | -0.1118 | 0.0293 | [-0.17, -0.06] |   221.42 | 1.000 | 45662.0000
## scramble3   | -0.1710 | 0.0293 | [-0.23, -0.11] | 1.35e+04 | 1.000 | 45983.0000
## yrs_mus_exp |  0.0002 | 0.0026 | [-0.01,  0.01] |    0.026 | 1.000 | 51337.0000
```

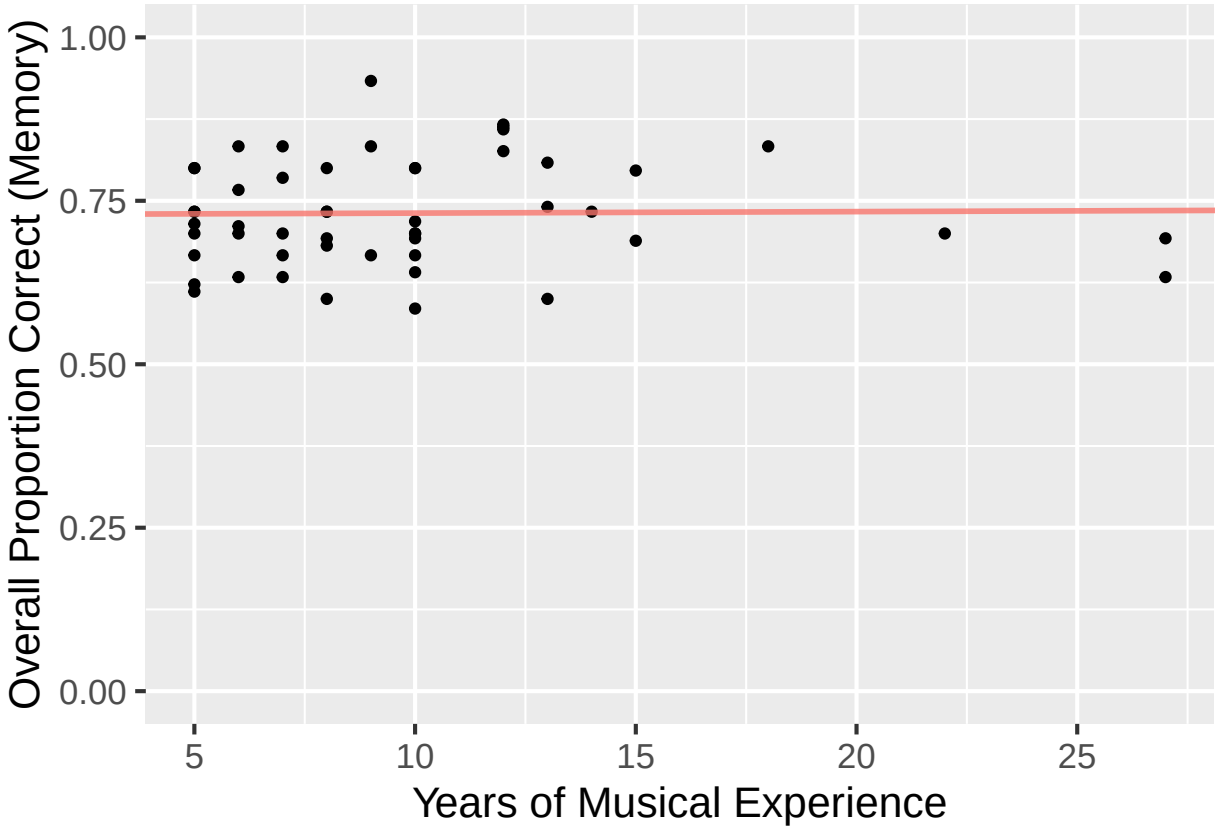
```
yrs_null_BF <- describe_posterior(years_mus,
                                  estimate = "median", dispersion = TRUE,
                                  ci = .95, ci_method = "HDI",
                                  test = c("bayes_factor"))
print(yrs_null_BF, digits = 4)
```

```
## Summary of Posterior Distribution
##
## Parameter   | Median |    MAD |        95% CI |       BF |  Rhat |        ESS
## ----------------------------------------------------------------------------
## (Intercept) | 0.7289 | 0.0311 | [ 0.67, 0.79] | 4.72e+26 | 1.000 | 66269.0000
## yrs_mus_exp | 0.0002 | 0.0028 | [-0.01, 0.01] |    0.028 | 1.000 | 68784.0000
```

Figure S1A

```
yrs_exp %>%
  group_by(exp_subject_id, yrs_mus_exp) %>%
  summarize(mean_acc = mean(accuracy)) %>%
  ggplot(aes(yrs_mus_exp, mean_acc)) +
  geom_point() +
  geom_abline(intercept = yrs_null_BF$Median[1], slope = yrs_null_BF$Median[2],
              color = '#F8766D', linewidth = 1, alpha = 0.8) +
  xlab('Years of Musical Experience') +
  ylab('Overall Proportion Correct (Memory)') +
  scale_x_continuous(breaks = seq(5,30,5)) +
  scale_y_continuous(breaks = seq(0, 1, 0.1)) +
  ylim(0,1) +
  theme_gray(base_size = 16)
```

```
## `summarise()` has grouped output by 'exp_subject_id'. You can override using
## the `.groups` argument.
## Scale for y is already present. Adding another scale for y, which will replace
## the existing scale.
```




```
ggsave('../figures/FigS1A_memory.png', width = 5, height = 5)
```