Supplement to Action Effects on Visual Perception of Distances: A Multilevel Bayesian Meta-Analysis

Lisa Molto[1], Ladislas Nalborczyk[1,2], Richard Palluel-Germain[1], & Nicolas Morgado[3]

[1] Univ. Grenoble Alpes, CNRS, LPNC, 38000 Grenoble, France
[2] Department of Experimental Clinical and Health Psychology, Ghent University
[3] Univ. Paris Nanterre, Centre de Recherche sur le Sport et le Mouvement, Nanterre, France

Contents

Correspondence concerning this article should be addressed to Lisa Molto, Laboratoire de Psychologie et Neurocognition, Univ. Grenoble Alpes, 1251 avenue centrale, 38058 Grenoble Cedex 9, France. E-mail: lisa.molto@univ-grenoble-alpes.fr

1 The meta-analytic model

Let y_{ij} be the effect size of the ith study in the jth article. Then, the 3-levels meta-analytic model can be written as:

$$\begin{aligned} y_{ij} &\sim \text{Normal}(\mu_{ij}, \sigma_{ij}) \\ \mu_{ij} &= \alpha + \alpha_{article[j]} + \alpha_{study[ij]} \\ \alpha_{study[ij]} &\sim \text{Normal}(0, \tau_s) \\ \alpha_{article[j]} &\sim \text{Normal}(0, \tau_a) \end{aligned}$$

Where σ^2_{ij} is the known sampling variance of the ith study in the jth article and α is the population effect size. The index $\alpha_{study[ij]}$ indicates the intercept corresponding to study i in article j (which is the average effect size in this study), and $\alpha_{article[j]}$ indicates the intercept for article j (which is the average effect size in this article). In addition to the sampling variance, there are two other sources of variation: the variance of the effect between studies $\text{Var}(\alpha_{study}) = \tau^2_s$ (level-2), and the variance of the effect between articles $\text{Var}(\alpha_{article}) = \tau^2_a$ (level-3).

1.1 Fitting the model

We can then write the full Bayesian model, including the priors for α (the intercept) and the variance components. The intercept of the model estimates the overall effect and is given an midly informative normal prior. The variance components τ_s and τ_a have midly informative Half-Cauchy priors, ensuring that very large values (which are implausible for the scale of y_{ij}), receive less prior weight.

$$\begin{aligned} y_{ij} &\sim \text{Normal}(\mu_{ij}, \sigma_{ij}) \\ \mu_{ij} &= \alpha + \alpha_{article[j]} + \alpha_{study[ij]} \\ \alpha_{study[ij]} &\sim \text{Normal}(0, \tau_s) \\ \alpha_{article[j]} &\sim \text{Normal}(0, \tau_a) \\ \alpha &\sim \text{Normal}(0, 1) \\ \tau_s, \tau_a &\sim \text{HalfCauchy}(0, 0.1) \end{aligned}$$

This model can easily be fitted using the `brms` package (Bürkner, 2017), with an lme4-like syntax.

```
library(tidyverse)
library(brms)

# setting the seed for reproducibility
set.seed(123)

# defining the priors
prior1 <- c(
    prior(normal(0, 1), coef = intercept),
```

```
    prior(cauchy(0, 0.1), class = sd)
    )

bmod1 <- brm(
    g | se(sqrt(vi) ) ~ 0 + intercept + (1|article) + (1|study),
    data = data,
    prior = prior1,
    sample_prior = FALSE,
    save_all_pars = TRUE,
    chains = 4,
    warmup = 5000,
    iter = 20000,
    cores = parallel::detectCores(),
    control = list(adapt_delta = .99)
    )
```

1.2 Retrieving the estimates

We then extract the posterior mean and 95% credible intervals of the previous model using the `tidy()` function of the `broom` package (Robinson, 2017).

```
library(broom)
tidy(bmod1, parameters = c("^b_", "^sd_"), prob = 0.95)
```

```
##                     term  estimate  std.error      lower     upper
## 1            b_intercept 0.4599480 0.12636799 0.22281339 0.7169685
## 2 sd_article__Intercept 0.4777046 0.12306150 0.25757458 0.7428181
## 3   sd_study__Intercept 0.1165465 0.06624345 0.01716537 0.2780264
```

We can interpret the intercept estimate by saying that the most credible value of the effect size is 0.46, and that there is a 95% probability (given the data and the model) that the population effect size lies in the [0.22, 0.72] interval.

1.3 Leave-one-out analysis

We then use leave-one-out analyses to investigate the influence of single studies on the obtained meta-analytic average effect. Basically, we fit again the main model by using all studies but one, doing this for all studies.

```
article_names <- sort(unique(data$article) )
bmods <- setNames(vector("list", length(article_names) ), article_names)

for (i in seq_along(article_names) ) {

  print(article_names[i])
```

```
  subdata <- droplevels(subset(data, article != article_names[i]) )

  capture.output({bmods[[i]] <- update(bmod1, newdata = subdata)})

}
```

Below we extract all the computed intercepts and report the `min` and `max` values of these intercepts, as an indication of the *robustness* of the main estimate.

```
intercepts_LOO <- as.numeric(unlist(lapply(bmods, function(x) brms::fixef(x)[1]) ) )
range(intercepts_LOO)
```

Inspection of these results reveals that Lessard et al. (2009)' is strongly deviant from other studies, with a reported effect size of g = 2.42. In the following, all analyses are carried out without this study.

```
# removing Lessard et al. (2009)
data %<>% filter(authors != "Lessard et al.")
data.m %<>% filter(authors != "Lessard et al.")

# fitting the model again
bmod1 <- brm(
    g | se(sqrt(vi) ) ~ 0 + intercept + (1|article) + (1|study),
    data = data,
    prior = prior1,
    sample_prior = FALSE,
    save_all_pars = TRUE,
    chains = 4,
    warmup = 5000,
    iter = 20000,
    cores = parallel::detectCores(),
    control = list(adapt_delta = .99)
    )
```

We then extract the posterior mean and 95% credible intervals of the updated general meta-analytic model.

```
tidy(bmod1, parameters = c("^b_", "^sd_"), prob = 0.95)
```

```
##                    term  estimate  std.error      lower     upper
## 1           b_intercept 0.2893580 0.07819671 0.15809003 0.4634606
## 2 sd_article__Intercept 0.1843994 0.10139793 0.01307961 0.4001296
## 3   sd_study__Intercept 0.1261737 0.06150120 0.02246054 0.2618341
```

We can interpret the intercept estimate by saying that the most credible value of the effect size is 0.29, and that there is a 95% probability (given the data and the model) that the population effect size lies in the [0.16, 0.46] interval.

1.4 Hypothesis testing

We can test the hypothesis that the intercept is equal to 0 by comparing a model with the intercept and a model without the intercept (i.e., with the intercept value fixed to 0). We compare these models using the `bayes_factor()` method (that uses the `bridgesampling` package, Gronau & Singmann, 2017).

```
prior0 <- prior(cauchy(0, 0.1), class = sd)

bmod0 <- brm(
    g | se(sqrt(vi) ) ~ 0 + (1|article) + (1|study),
    data = data,
    prior = prior0,
    sample_prior = FALSE,
    save_all_pars = TRUE,
    chains = 4,
    warmup = 5000,
    iter = 20000,
    cores = parallel::detectCores(),
    control = list(adapt_delta = .99)
    )

bf_intercept <- bayes_factor(
    bmod0, bmod1,
    repetitions = 1e2, cores = parallel::detectCores()
    )
```

The Bayes Factor is given by `bf_intercept$bf_median_based` (median estimate based on 100 simulations) and is approximately equal to $BF_{10} = \frac{1}{BF_{01}} = 281.14$.

1.5 Forest plot of the main model

The next figure depicts the estimates of the above model, where densities represent the estimation of the model (i.e., the posterior distibution), along with its mean and the 95% credible interval.

```
# installing (if needed) and loading the brmstools package
if(!require(brmstools) ) devtools::install_github("mvuorre/brmstools")
library(brmstools)

# sourcing a slightly modified version of Matti Vuorre's function
source(here("code", "forest.R") )

# fitting the main model (with the intercept)
prior1_forest <- c(
    prior(normal(0, 1), class = Intercept),
```

```
    prior(cauchy(0, 0.1), class = sd)
    )

bmod1_forest <- brm(
    g | se(sqrt(vi) ) ~ 1 + (1|article) + (1|study),
    data = data,
    prior = prior1_forest,
    sample_prior = FALSE,
    save_all_pars = TRUE,
    chains = 4,
    warmup = 5000,
    iter = 20000,
    cores = parallel::detectCores(),
    control = list(adapt_delta = .99)
    )

# making the forest plot
forest(
    bmod1_forest, grouping = "article", theme_forest = FALSE,
    show_data = TRUE, sort = FALSE, av_name = "Overall effect size"
    ) +
    theme_ipsum(
        base_family = "Helvetica",
        base_size = 10, plot_title_size = 12, axis_text_size = 9
        ) +
    xlab("Effect size (Hedge's g)") +
    ylab("") +
    xlim(-0.5, 2)
```

1.6 Estimation by constraint manipulation

We previously provided an estimate of the global effect size of action constraint on distance perception. Below, we estimate the effect size based on the action constraint manipulation by including `manipulation` as a categorical predictor in the model. We have three categories of manipulation that are: tool-use, weight, and effort. In this model the intercept represents the average effect size corresponding to the `Effort` manipulation, and the βs for `Tool-use` and `Weight` represent deviations from this condition.

```
data.constraint <- data %>% filter(manipulation != "other")

prior2 <- c(
    prior(normal(0, 1), coef = "intercept"),
    prior(normal(0, 1), class = b),
    prior(cauchy(0, 0.1), class = sd)
```

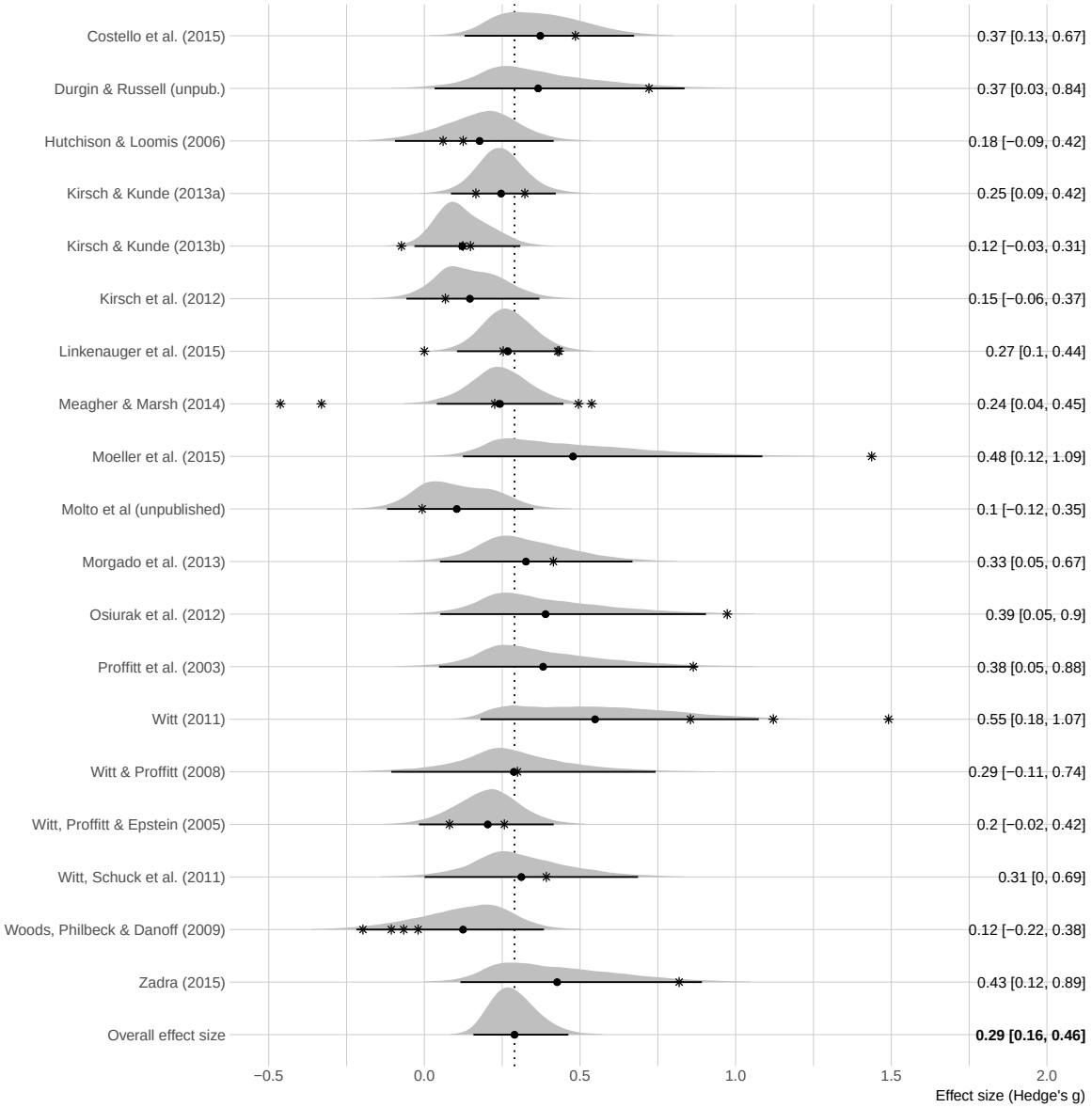

Figure 1. Forest plot of effect sizes. The densities represent the estimation of the model: the posterior distribution, along with its mean and 95% credible interval. Raw data (for each experiment) are represented by the stars.

```
    )

bmod_manip <- brm(
    g | se(sqrt(vi) ) ~ 0 + intercept + manipulation + (1|article) + (1|study),
    data = data,
    prior = prior2,
    sample_prior = TRUE,
    warmup = 5000,
    iter = 20000,
    cores = parallel::detectCores(),
    control = list(adapt_delta = .99)
    )
```

Below we retrieve samples from the posterior distribution using the `posterior_samples()` function. Then, we can use these posterior samples to compare each pair of conditions.

```
# retrieving the posterior samples
post_manip <- posterior_samples(bmod_manip, pars = "^b_")

# difference between effort and tool-use
contrast1 <- post_manip[, 1]  - (post_manip[, 1] + post_manip[, 2])

# difference between effort and weight
contrast2 <- post_manip[, 1]  - (post_manip[, 1] + post_manip[, 3])

# difference between tool-use and weight
contrast3 <- (post_manip[, 1] + post_manip[, 2]) - (post_manip[, 1] + post_manip[, 3])
```

We then compute Bayes factors for each contrast using the `hypothesis()` method. This analysis reveals moderate evidence for an absence of difference between `Tool-use` and `Weight` (β = 0.27, 95% CrI [-0.23, 0.77, BF_{01} = 3.10), moderate evidence for an absence of difference between `Effort` and `Weight` (β = 0.19, 95% CrI [-0.26, 0.65, BF_{01} = 3.16), and moderate evidence for an absence of difference between `Effort` and `Tool-use` (β = -0.07, 95% CrI [-0.43, 0.30, BF_{01} = 5.24). We also test the null hypothesis for each constraint manipulation. This reveals moderate evidence for the hypothesis of no effect of weight manipulation (β = 0.13, 95% CrI [-0.26, 0.55], BF_{01} = 6.16). However, this analysis also reveals moderate to strong evidence for an effect of the AC manipulation (tool-use: BF_{10} = 3.81 and effort: BF_{10} = 10.45).

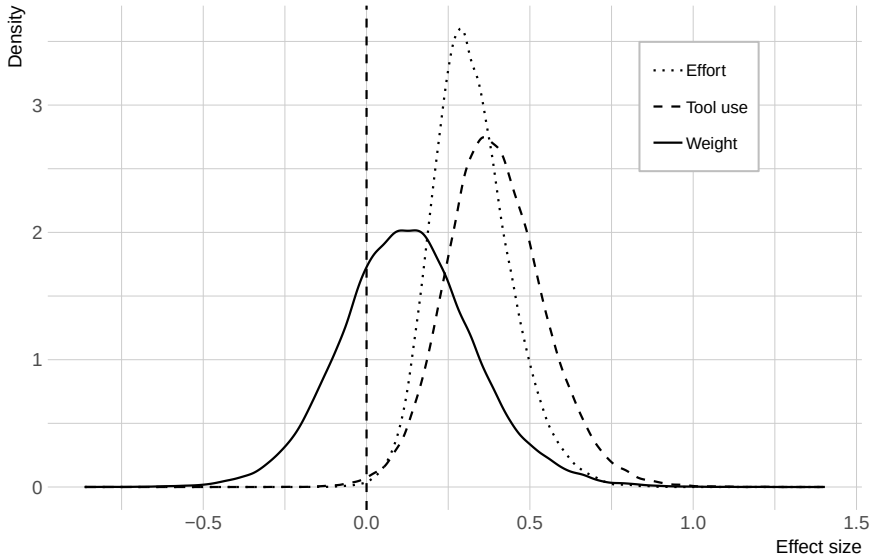

Figure 2. Posterior distribution of effect size by constraint manipulation.

2 Moderators analyses

We then fit one model by moderator (meta-regression models) using contrast codes for the research `design`, the `motor` intention, and `space`.

```
# removing "size" measure
data.measure <- data.m %>% filter(Measure != "Size")

# contrast coding the moderators
data$design.c <- ifelse(data$Design == 0, -0.5, 0.5)
data.m$motor.c <- ifelse(data.m$motor_i == 0, -0.5, 0.5)
data$space.c <- ifelse(data$space == "proche", -0.5, 0.5)
```

2.1 Research design

We fit a new model including `design` as a contrast-coded (-0.5, 0.5) predictor and assign it a midly informative normal prior.

```
bmod_design <- brm(
    g | se(sqrt(vi) ) ~ 0 + intercept + design.c + (1|article) + (1|study),
    data = data,
    prior = prior2,
    sample_prior = FALSE,
    save_all_pars = TRUE,
    chains = 4,
    warmup = 5000,
    iter = 20000,
    cores = parallel::detectCores(),
```

```
    control = list(adapt_delta = .99)
    )
```

Estimations from this model can be retrieved using the `tidy()` function, as previously.

```
(bmod_design_est <- tidy(bmod_design, parameters = c("^b_", "^sd_"), prob = 0.95) )
```

```
##                    term   estimate  std.error       lower       upper
## 1           b_intercept  0.3365955 0.07679124  0.20055317 0.502501136
## 2            b_design.c -0.2607505 0.13759405 -0.54082671 0.005648886
## 3 sd_article__Intercept  0.1723962 0.09069850  0.01512193 0.365961446
## 4   sd_study__Intercept  0.1096473 0.05503013  0.01658359 0.233728248
```

We can test the hypothesis of no difference between the two conditions following the same strategy as previously by comparing the previous model (`bmod_design`) to the intercept-only model.

```
bf_design <- bayes_factor(
    bmod1, bmod_design,
    repetitions = 1e2, cores = parallel::detectCores()
    )
```

There is only anecdotal evidence for an absence of difference between the two conditions (β = -0.26, 95% CrI [-0.54, 0.01], BF_{01} = 1.04).

2.2 Measures

We then fit a second model, including `measures` as a categorical predictor. One advantage of the Bayesian approach is that we can compare conditions (i.e., the different levels of the `measure` factor) directly from the joint posterior distribution by computing the posterior distribution of the difference.

```
bmod_measure <- brm(
    g | se(sqrt(vi) ) ~ 0 + intercept + Measure + (1|article) + (1|study),
    data = data.measure,
    prior = prior2,
    sample_prior = TRUE,
    warmup = 5000,
    iter = 20000,
    cores = parallel::detectCores(),
    control = list(adapt_delta = .99)
    )
```

```
(bmod_measure_est <- tidy(bmod_measure, parameters = c("^b_", "^sd_"), prob = 0.95) )
```

```
##                    term   estimate  std.error        lower     upper
## 1           b_intercept 0.21248606 0.15262114 -0.084067579 0.5172644
## 2       b_MeasureVerbal 0.02308512 0.16067934 -0.289434887 0.3446412
## 3           b_MeasureVm 0.08985221 0.16358243 -0.219500987 0.4223592
## 4 sd_article__Intercept 0.16751815 0.11033420  0.007429043 0.4075180
## 5   sd_study__Intercept 0.19733426 0.07550467  0.060562809 0.3552492
```

In this model the intercept represents the condition `MeasureAction`, and the βs for `MeasureVerbal` and `MeasureVm` represent deviations from this condition. Below we retrieve samples from the posterior distribution using the `posterior_samples()` function.

```
post <- posterior_samples(bmod_measure, pars = "^b_")
head(post)
```

```
##   b_intercept b_MeasureVerbal b_MeasureVm
## 1  0.29010040     -0.03829614  0.06234317
## 2  0.01358407     -0.05568620  0.26858131
## 3  0.22265255     -0.02671598 -0.08573913
## 4 -0.12663250      0.36230912  0.34300500
## 5  0.25094556     -0.05162500  0.23697637
## 6  0.25326721      0.06969551  0.23901304
```

Then, we can use these posterior samples to compare the conditions with each other.

```
# difference between verbal and vm
c1 <- (post[, 1] + post[, 2]) - (post[, 1] + post[, 3])

# difference between verbal and action
c2 <- (post[, 1] + post[, 2]) - post[, 1]

# difference between action and vm
c3 <- post[, 1] - (post[, 1] + post[, 3])
```

We can plot the posterior distribution corresponding to each *contrast*, using the `BEST` package (Kruschke & Meredith, 2017). Below, we plot the contrast `c1`, which represents the comparison of the `verbal` and `vm` conditions.

```
library(BEST)
par(cex = 0.75, cex.lab = 0.75)
plotPost(c1, credMass = 0.95, compVal = 0, col = "#b3cde0")
```

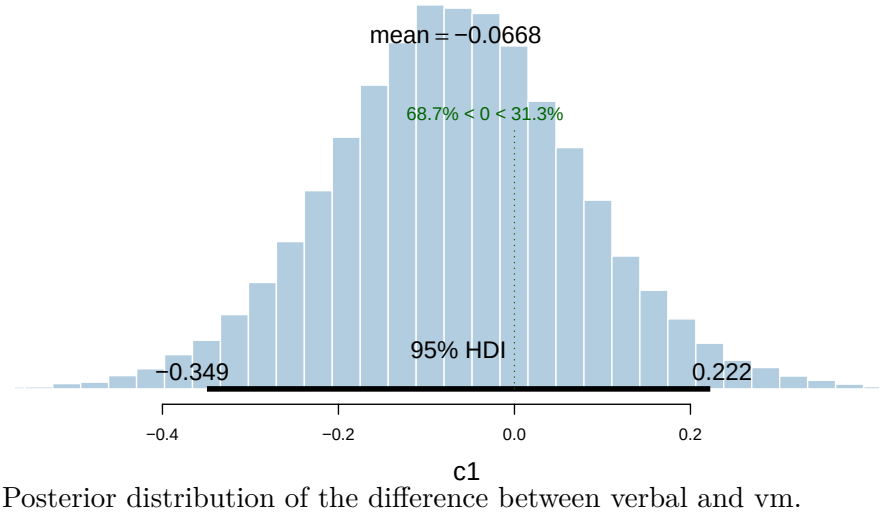

Figure 3. Posterior distribution of the difference between verbal and vm.

We can then compute the Bayes Factor for this difference using the `hypothesis()` function.

```
hypothesis(
    bmod_measure, "(intercept + MeasureVerbal) = (intercept + MeasureVm)",
    seed = 123
    )
```

```
## Hypothesis Tests for class b:
##                 Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio
## 1 ((intercept+Measu... = 0    -0.07      0.14    -0.35     0.22       9.08
##   Post.Prob Star
## 1       0.9
## ---
## '*': The expected value under the hypothesis lies outside the 95%-CI.
## Posterior probabilities of point hypotheses assume equal prior probabilities.
```

We can then follow the same strategy for the two other contrasts, `c2` and `c3`, by plotting the posterior distribution of the difference between the two conditions.

```
par(cex = 0.75, cex.lab = 0.75)
plotPost(c2, credMass = 0.95, compVal = 0, col = "#b3cde0")
```

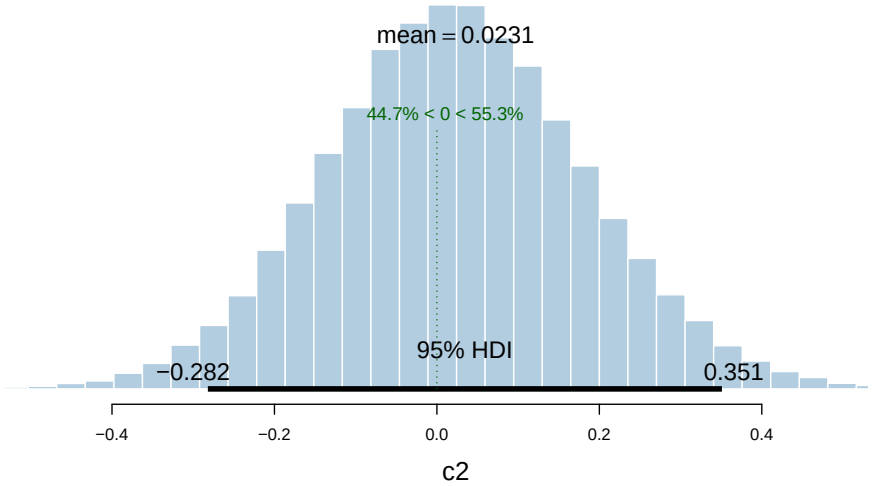

Figure 4. Posterior distribution of the contrast.

```
plotPost(c3, credMass = 0.95, compVal = 0, col = "#b3cde0")
```

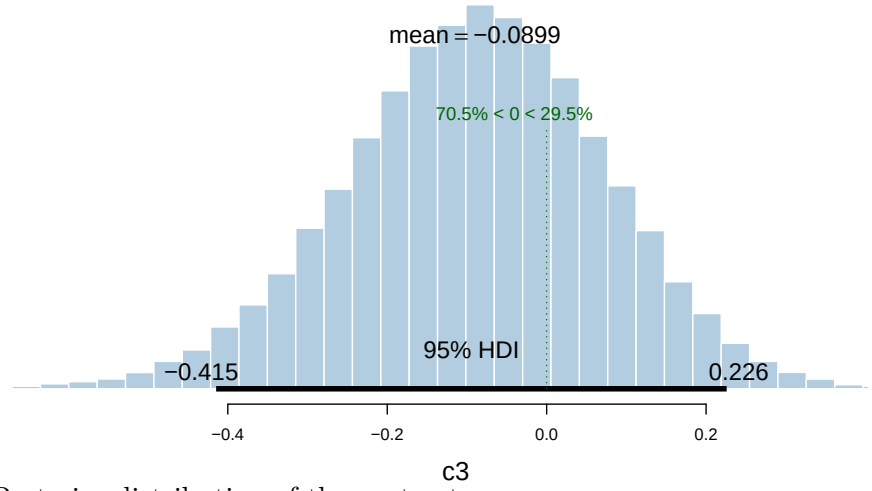

Figure 5. Posterior distribution of the contrast.

We then compute the Bayes Factor for these differences using the `hypothesis()` function, as previously.

```
hypothesis(
    bmod_measure, "(intercept) = (intercept + MeasureVerbal)",
    seed = 123
    )
```

```
## Hypothesis Tests for class b:
##                 Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio
## 1 ((intercept))-((i... = 0    -0.02      0.16    -0.34     0.29       6.28
```

```
##   Post.Prob Star
## 1      0.86
## ---
## '*': The expected value under the hypothesis lies outside the 95%-CI.
## Posterior probabilities of point hypotheses assume equal prior probabilities.
```

```
hypothesis(
    bmod_measure, "(intercept) = (intercept + MeasureVm)",
    seed = 123
    )
```

```
## Hypothesis Tests for class b:
##                 Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio
## 1 ((intercept))-((i... = 0    -0.09      0.16    -0.42     0.22       5.55
##   Post.Prob Star
## 1      0.85
## ---
## '*': The expected value under the hypothesis lies outside the 95%-CI.
## Posterior probabilities of point hypotheses assume equal prior probabilities.
```

2.3 Motor intention

As previously, we can fit a new model including `motor intention` as a contrast-coded categorical predictor with a midly informative normal prior.

```
bmod_motor <- brm(
    g | se(sqrt(vi) ) ~ 0 + intercept + motor.c + (1|article) + (1|study),
    data = data.m,
    prior = prior2,
    save_all_pars = TRUE,
    chains = 4,
    warmup = 5000,
    iter = 20000,
    cores = parallel::detectCores(),
    control = list(adapt_delta = .99)
    )
```

```
(bmod_motor_est <- tidy(bmod_motor, parameters = c("^b_", "^sd_"), prob = 0.95) )
```

```
##                    term   estimate  std.error       lower     upper
## 1           b_intercept 0.25538755 0.09374274  0.08907247 0.4562718
## 2             b_motor.c 0.06930065 0.14941719 -0.23000842 0.3606807
## 3 sd_article__Intercept 0.20345941 0.11063563  0.01432974 0.4365166
## 4   sd_study__Intercept 0.15321818 0.06982372  0.03348667 0.3031598
```

We can then test whether the difference between the two conditions is equal to zero, following the same strategy as previously.

```
bf_motor <- bayes_factor(
    bmod1, bmod_motor,
    repetitions = 1e2, cores = parallel::detectCores()
    )
```

There is strong evidence for an absence of difference between the two conditions (β = 0.07, 95% CrI [-0.23, 0.36], BF_{01} = 10,096.54).

2.4 Distance from target

We fit below a new model including target presence (`space.c`) as a contrast-coded (-0.5, 0.5) categorical predictor, and assign it a midly informative normal prior.

```
bmod_space <- brm(
    g | se(sqrt(vi) ) ~ 0 + intercept + space.c + (1|article) + (1|study),
    data = data,
    prior = prior2,
    sample_prior = FALSE,
    save_all_pars = TRUE,
    chains = 4,
    warmup = 5000,
    iter = 20000,
    cores = parallel::detectCores(),
    control = list(adapt_delta = .99)
    )
```

Estimations of this model can be retrieved using the `tidy()` function.

```
(bmod_space_est <- tidy(bmod_space, parameters = c("^b_", "^sd_"), prob = 0.95) )
```

```
##                     term   estimate  std.error       lower     upper
## 1          b_intercept 0.31425196 0.08593040  0.16586409 0.5037808
## 2            b_space.c 0.09729085 0.15428004 -0.20127676 0.4150503
## 3 sd_article__Intercept 0.20681083 0.10472691  0.01852187 0.4265177
## 4   sd_study__Intercept 0.11866278 0.06073119  0.01946639 0.2573810
```

We can test whether the difference between the two conditions is equal to zero by computing a Bayes Factor using the `bayes_factor()` method.

```
bf_space <- bayes_factor(bmod1, bmod_space, repetitions = 1e2, cores = parallel::detectCore
```

There is only anecdotal evidence for an absence of difference between the two conditions (β = 0.10, 95% CrI [-0.20, 0.42], BF_{01} = 5.71). Below, we plot the marginal posterior distribution of the effect size by condition, for each moderator.

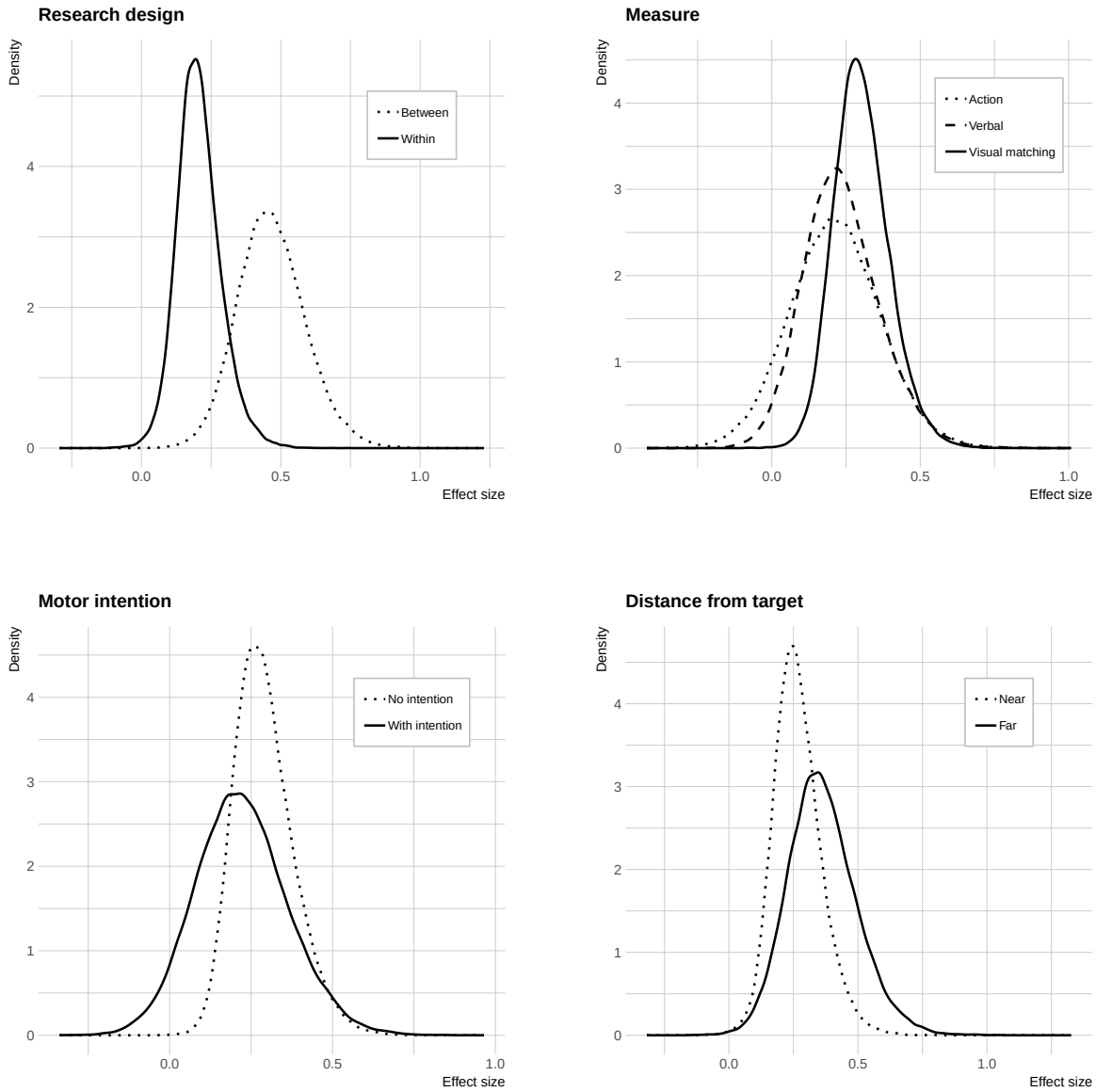

Figure 6. Posterior distribution of the effect size by condition.

3 Additional analyses

3.1 Publication bias

If there is no publication bias, the funnel plot should look symmetric with outcomes dispersed equally on both sides of the average effect size. However, if studies with null results are missing, the funnel plot might look asymmetric with studies on the left side of the mean estimate missing.

```
library(metaviz)
library(metafor)
```

```
# create the funnel plot from a metafor model (for convenience)
res <- rma(yi = g, vi, data = data, measure = "GEN")

# customised ggplot2 funnel plot
source(here("code", "funnel_viz2.R") )

viz_funnel2(
    x = res, method = "REML",
    contours = TRUE, sig_contours = FALSE,
    contours_col = "Greys", xlab = "Observed outcome"
    ) +
    theme_ipsum(
        base_family = "Helvetica",
        base_size = 10, plot_title_size = 12, axis_text_size = 9
        )
```

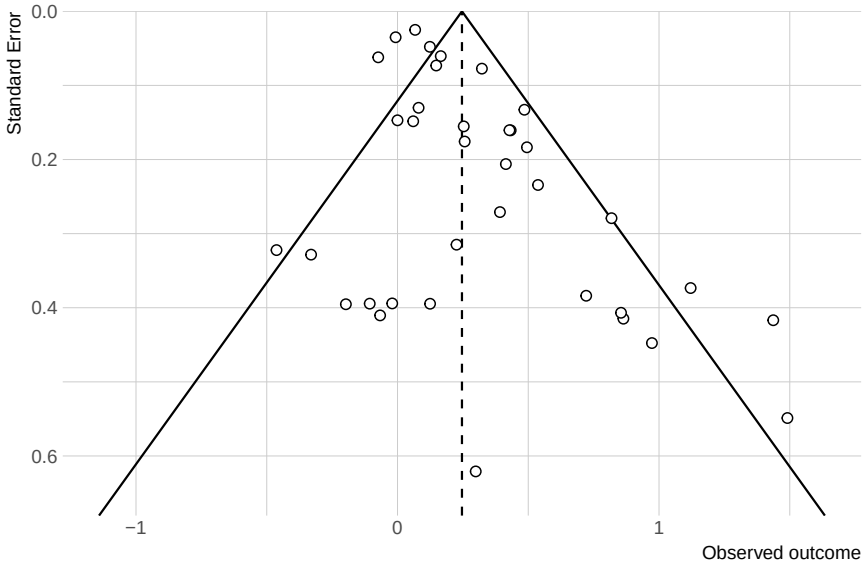

Figure 7. Funnel plot with effect sizes on x-axis and standard errors on the y-axis.

Below, we make a contour-enhanced funnel plot, allowing to distinguish patterns of significance in published studies.

```
viz_funnel(
    x = res, method = "REML",
    contours = FALSE, sig_contours = TRUE, detail_level = 1,
    contours_col = "Greys",
    xlab = "Observed outcome"
    ) +
    geom_vline(xintercept = 0, linetype = 2) +
```

```
    theme_ipsum(
        base_family = "Helvetica",
        base_size = 10, plot_title_size = 12, axis_text_size = 9
        )
```

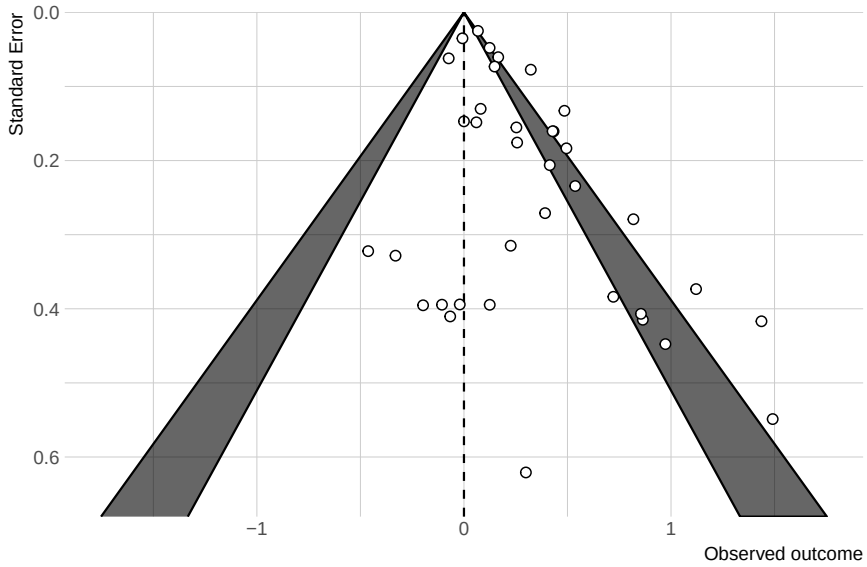

Figure 8. Contour-enhanced funnel plot with effect sizes on x-axis and standard errors on the y-axis. Ranges of p-values from the experiments are indicated by the shaded regions. White region inside the funnel corresponds to p-values greater than .10, the gray-shaded region corresponds to p-values between .05 and .01, and the region outside of the funnel corresponds to p-values below .01.

3.2 P-curve

P-curve analyses test whether a set of p-values supports the existence of an effect. If there is no effect, p-values should be uniformly distributed, whereas if there is an effect, p-values distribution should be right-skewed, with more p-values close to .01 than .05. The observed distribution of p-values from our database is close to a distribution that would be expected with a *true* non-null effect, although it does not differ significantly from the distribution of p-values we would expect for a small effect size and a power of 33%. Moreover, the uptick of the observed p-curve at p-values of .04 and .05 might indicate data snooping (e.g., p-hacking, QRPs).

```
source(here("code", "pcurve", "pcurve.R") )
knitr::include_graphics(here("code", "pcurve", "p_curve.pdf") )
```

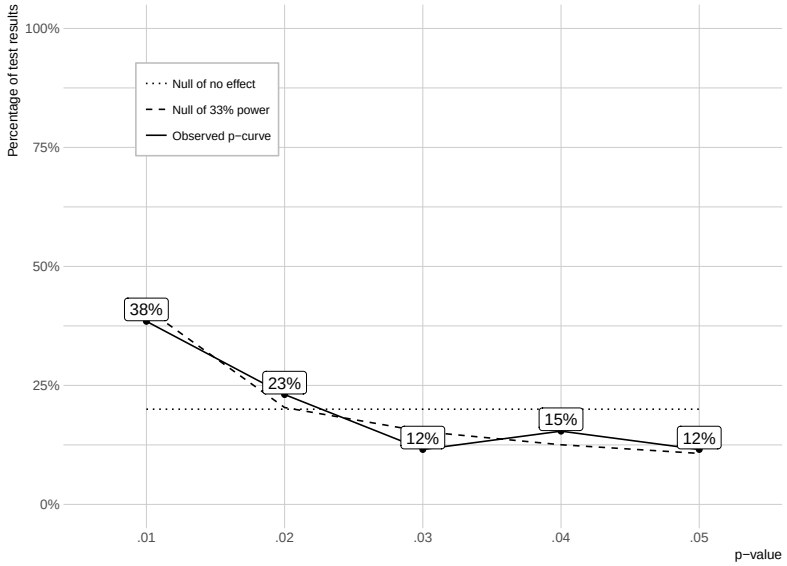

Figure 9. P-curve estimation of average power of individual studies: .85, 90% CI [.70, .94]. The observed p-curve includes 21 statistically significant ($p < .05$) results of which 17 were inferior to .025. There were 14 additional results entered but excluded from p-curve because they were superior to .05.

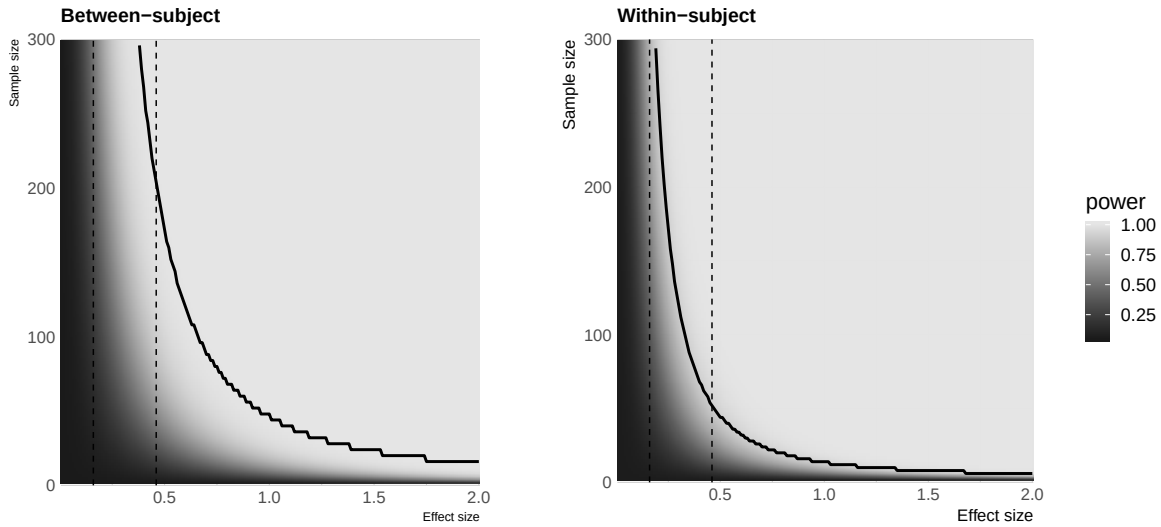

Figure 10. Statistical power as a function of sample size, effect size, and research design (left panel: between-subject design; right panel: within-subject design), with a Type 1 error rate (alpha level) fixed at .05. The interval between the dashed lines indicates the 95% most credible effect sizes as estimated by our model. The black curve corresponds to the recommended power (0.9).

3.3 Power analysis

Below, we illustrate the sample size needed to reach a given level of statistical power according to the AC effects of interest.

4 Session information

```
sessionInfo()
```

```
## R version 3.5.3 (2019-03-11)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Mojave 10.14
##
## Matrix products: default
## BLAS: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] brms_2.8.0          Rcpp_1.0.1          here_0.1
##  [4] knitr_1.22          readxl_1.3.1        papaja_0.1.0.9709
##  [7] magrittr_1.5        forcats_0.4.0       stringr_1.4.0
## [10] dplyr_0.8.1         purrr_0.3.2         readr_1.3.1
## [13] tidyr_0.8.3         tibble_2.1.1        ggplot2_3.1.1
## [16] tidyverse_1.2.1     patchwork_0.0.1     hrbrthemes_0.6.0
## [19] bridgesampling_0.6-0
##
## loaded via a namespace (and not attached):
##  [1] colorspace_1.4-1   ggridges_0.5.1     rsconnect_0.8.13
##  [4] rprojroot_1.3-2    markdown_0.9       base64enc_0.1-3
##  [7] rstudioapi_0.10    rstan_2.18.2       DT_0.5
## [10] mvtnorm_1.0-10     lubridate_1.7.4    xml2_1.2.0
## [13] extrafont_0.17     shinythemes_1.1.2  bayesplot_1.6.0
## [16] jsonlite_1.6       broom_0.5.2        Rttf2pt1_1.3.7
## [19] shiny_1.3.1        compiler_3.5.3     httr_1.4.0
## [22] backports_1.1.4    assertthat_0.2.1   Matrix_1.2-17
## [25] lazyeval_0.2.2     cli_1.1.0          later_0.8.0
## [28] htmltools_0.3.6    prettyunits_1.0.2  tools_3.5.3
## [31] igraph_1.2.4       coda_0.19-2        gtable_0.3.0
## [34] glue_1.3.1         reshape2_1.4.3     cellranger_1.1.0
## [37] nlme_3.1-139       extrafontdb_1.0    crosstalk_1.0.0
## [40] xfun_0.7           ps_1.3.0           rvest_0.3.3
## [43] mime_0.7           miniUI_0.1.1.1     gtools_3.8.1
## [46] zoo_1.8-5          scales_1.0.0       colourpicker_1.0
## [49] hms_0.4.2          promises_1.0.1     Brobdingnag_1.2-6
```

```
## [52] parallel_3.5.3     inline_0.3.15      shinystan_2.5.0
## [55] yaml_2.2.0         gridExtra_2.3      gdtools_0.1.8
## [58] loo_2.1.0.9000     StanHeaders_2.18.1 stringi_1.4.3
## [61] dygraphs_1.1.1.6   pkgbuild_1.0.3     rlang_0.3.4
## [64] pkgconfig_2.0.2    matrixStats_0.54.0 evaluate_0.13
## [67] lattice_0.20-38    rstantools_1.5.1   htmlwidgets_1.3
## [70] tidyselect_0.2.5   processx_3.3.0     plyr_1.8.4
## [73] bookdown_0.9.17    R6_2.4.0           generics_0.0.2
## [76] pillar_1.4.0       haven_2.1.0        withr_2.1.2
## [79] xts_0.11-2         abind_1.4-5        modelr_0.1.4
## [82] crayon_1.3.4       rmarkdown_1.12     grid_3.5.3
## [85] callr_3.2.0        threejs_0.3.1      digest_0.6.18
## [88] xtable_1.8-3       httpuv_1.5.1       stats4_3.5.3
## [91] munsell_0.5.0      shinyjs_1.0
```

5 References

Bürkner, P.-C. (2017). brms: An R package for bayesian multilevel models using Stan. *Journal of Statistical Software*, *80*(1), 1–28. doi:10.18637/jss.v080.i01

Gronau, Q. F., & Singmann, H. (2017). *Bridgesampling: Bridge sampling for marginal likelihoods and bayes factors*. Retrieved from https://CRAN.R-project.org/package=bridgesampling

Kruschke, J. K., & Meredith, M. (2017). *BEST: Bayesian estimation supersedes the t-test*. Retrieved from https://CRAN.R-project.org/package=BEST

Robinson, D. (2017). *Broom: Convert statistical analysis objects into tidy data frames*. Retrieved from https://CRAN.R-project.org/package=broom