Assignment 2 - Meta-analysis of pitch in schizophrenia

Elisius, Laurits, Tilde & Niels

15/12-2022

Assignment 2: meta-analysis

Questions to be answered

1. Simulate data to setup the analysis and gain insight on the structure of the problem. Simulate one dataset of 100 studies (n of participants should follow a normal distribution with mean of 20, sd of 10, but no fewer than 10 participants), with a mean effect size of 0.4, average deviation by study of .4 and measurement error of .8. The data you get should have one row per study, with an effect size mean and standard error. Build a proper bayesian model to analyze the simulated data. Then simulate publication bias (only some of the studies you simulate are likely to be published, which?), the effect of publication bias on your estimates (re-run the model on published studies, assess the difference), and discuss what this implies for your model. remember to use at least one plot to visualize your results. BONUS question: do a power/precision analysis: w this kind of sample sizes (participants) how many studies would you need to acquire good precision (e.g. .1 sd in the pop level estimate)

Question 1: Simulate data to setup the analysis and gain insight on the structure of the problem.

Data simulation (NV & TS)

```r
#Installing packages and setting seed for repreoducability
pacman::p_load(tidyverse, brms, bayesplot, rstanarm, msm, cmdstanr, gridExtra, metafor)
set.seed(123)
#parameter identification
EffectMean <- 0.4
StudySD <- 0.4
Error <- 0.8
#specifying number of studies
Studies <- 100
n_participants <- 20
n_participants_sd <- 10
min_participants <- 10

#simulating 100 studies
d <- tibble(
  Study = seq(Studies),
  Participants = round(msm::rtnorm(Studies, n_participants, n_participants_sd, lower = min_pa
rticipants), 0),
  StudyEffect = NA,
  EffectMu = NA,
  EffectError = NA
)

#simulating parameters for each study
for (i in seq(Studies)){
  d$StudyEffect[i] <- rnorm(1, EffectMean, StudySD)
  sampling <- rnorm(d$Participants[i], d$StudyEffect[i], Error)
  d$EffectMu[i] <- mean(sampling)
  d$EffectError[i] <- sd(sampling)/sqrt(d$Participants[i])
}

#setting priors
ma_p0 <- c(
  prior(normal(0, 0.3), class = Intercept),
  prior(normal(0, 0.3), class = sd))

ma_p1 <- c(
  prior(normal(0, 0.3), class = Intercept),
  prior(normal(0, 0.3), class = sd))

#setting models
ma_f0 <- bf(EffectMu | se(EffectError) ~ 1 + (1|Study))


#getting priors for the models
#priors for ma_f0
get_prior(ma_f0,
          data = d,
          family = gaussian)
```

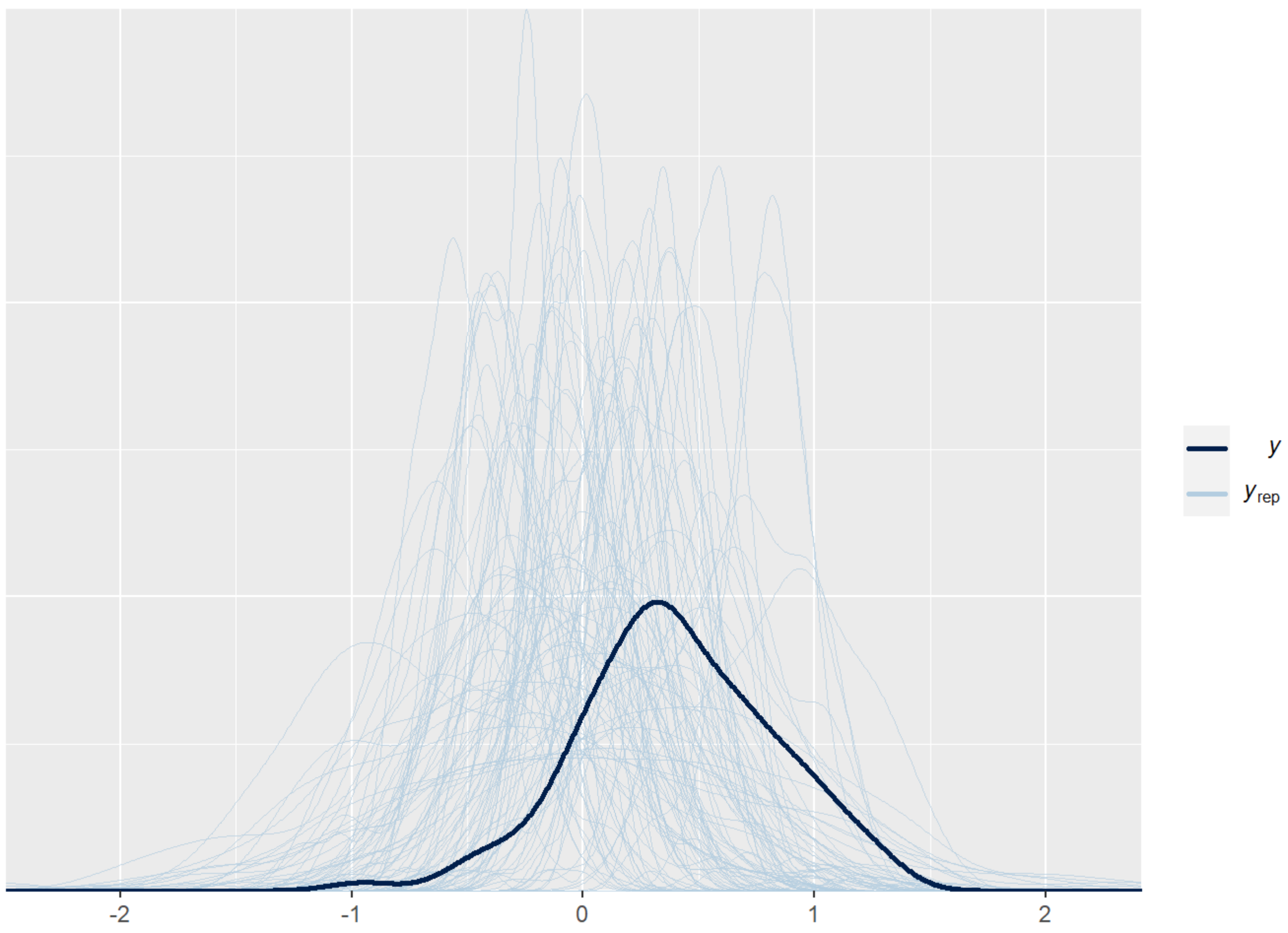
```
##                    prior     class      coef group resp dpar nlpar lb ub
##  student_t(3, 0.4, 2.5) Intercept
##    student_t(3, 0, 2.5)        sd                                  0
##    student_t(3, 0, 2.5)        sd           Study                  0
##    student_t(3, 0, 2.5)        sd Intercept Study                  0
##        source
##       default
##       default
##  (vectorized)
##  (vectorized)
```

```
#running the models
ma_m0_prior <- brm(
  ma_f0,
  data = d,
  family = gaussian,
  prior = ma_p0,
  sample_prior = "only",
  backend = "cmdstanr",
  threads = threading(2),
  chains = 2,
  cores = 8,
  control = list(
    adapt_delta = 0.9,
    max_treedepth = 20))
```

```
## Running MCMC with 2 chains, at most 8 in parallel, with 2 thread(s) per chain...
##
## Chain 1 Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1 Iteration:  100 / 2000 [  5%]  (Warmup)
## Chain 1 Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1 Iteration:  300 / 2000 [ 15%]  (Warmup)
## Chain 1 Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1 Iteration:  500 / 2000 [ 25%]  (Warmup)
## Chain 1 Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1 Iteration:  700 / 2000 [ 35%]  (Warmup)
## Chain 1 Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1 Iteration:  900 / 2000 [ 45%]  (Warmup)
## Chain 1 Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1 Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2 Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2 Iteration:  100 / 2000 [  5%]  (Warmup)
## Chain 2 Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2 Iteration:  300 / 2000 [ 15%]  (Warmup)
## Chain 2 Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2 Iteration:  500 / 2000 [ 25%]  (Warmup)
## Chain 2 Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2 Iteration:  700 / 2000 [ 35%]  (Warmup)
## Chain 2 Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2 Iteration:  900 / 2000 [ 45%]  (Warmup)
## Chain 2 Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2 Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2 Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 1 Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 1 Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1 Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 1 Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1 Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 1 Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1 Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 1 Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1 Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 1 Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2 Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2 Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 2 Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2 Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 2 Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2 Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 2 Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2 Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 2 Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1 finished in 0.3 seconds.
## Chain 2 finished in 0.3 seconds.
##
## Both chains finished successfully.
## Mean chain execution time: 0.3 seconds.
## Total execution time: 0.4 seconds.
```

Creating publication bias (NV)

```
#symmetric prior bias
pp_check(
  ma_m0_prior, ndraws = 100)
```



```
#including publication bias
d1 <- d %>% mutate(
  Published = NA,
  PublishedPos = NA)

#loops through all the studies
for (i in seq(Studies)){
  #checks if the absolute value of the specific study's effectMU minus two standard
    #deviations is above 0. If it is there is a 95% chance that the study is published, if no
t       only a 5% chance of being published.
  d1$Published[i] <- ifelse(
    abs(d$EffectMu[i]) - (2*d$EffectError[i]) > 0,
    rbinom(1, 1, .9), rbinom(1, 1, .1))
  d1$PublishedPos[i] <- ifelse(
    #does the same as the above ifelse function but the study's effectMU also has to be
#positive.
      abs(d$EffectMu[i]) - (2*d$EffectError[i]) > 0 & d$EffectMu[i] > 0,
    rbinom(1, 1, .9), rbinom(1, 1, .1))
  }


#updating the models
ma_m0_all <- brm(
  ma_f0,
  data = d1,
  save_pars = save_pars(all = T),
  family = gaussian,
  prior = ma_p0,
  sample_prior = T,
  backend = "cmdstanr",
  threads = threading(2),
  chains = 2,
  cores = 8,
  control = list(
    adapt_delta = 0.9,
    max_treedepth = 20
  ))
```

```
## Running MCMC with 2 chains, at most 8 in parallel, with 2 thread(s) per chain...
##
## Chain 1 Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2 Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1 Iteration:  100 / 2000 [  5%]  (Warmup)
## Chain 2 Iteration:  100 / 2000 [  5%]  (Warmup)
## Chain 2 Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1 Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1 Iteration:  300 / 2000 [ 15%]  (Warmup)
## Chain 1 Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1 Iteration:  500 / 2000 [ 25%]  (Warmup)
## Chain 2 Iteration:  300 / 2000 [ 15%]  (Warmup)
## Chain 2 Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2 Iteration:  500 / 2000 [ 25%]  (Warmup)
## Chain 2 Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2 Iteration:  700 / 2000 [ 35%]  (Warmup)
## Chain 1 Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1 Iteration:  700 / 2000 [ 35%]  (Warmup)
## Chain 1 Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1 Iteration:  900 / 2000 [ 45%]  (Warmup)
## Chain 1 Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1 Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2 Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2 Iteration:  900 / 2000 [ 45%]  (Warmup)
## Chain 2 Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2 Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1 Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 1 Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2 Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 2 Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2 Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 1 Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 1 Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1 Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 2 Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2 Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 2 Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1 Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1 Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 1 Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2 Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 2 Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2 Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 1 Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 1 Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2 Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1 finished in 0.9 seconds.
## Chain 2 finished in 0.9 seconds.
##
## Both chains finished successfully.
## Mean chain execution time: 0.9 seconds.
## Total execution time: 1.0 seconds.
```

```r
ma_m0_pub <- update(
  ma_m0_all, newdata = subset(d1, Published == 1))
```

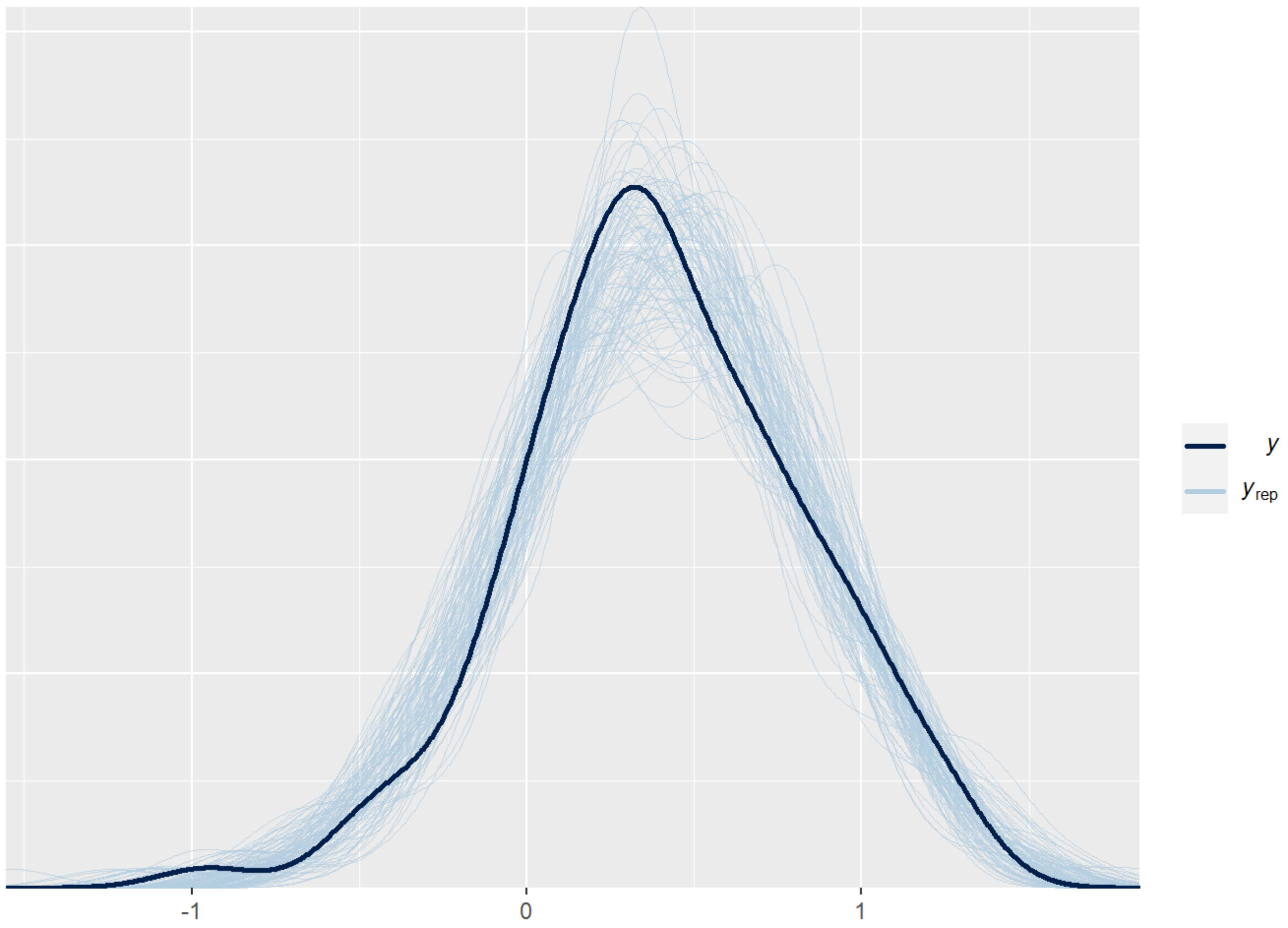
```
## Running MCMC with 2 sequential chains, with 2 thread(s) per chain...
##
## Chain 1 Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1 Iteration:  100 / 2000 [  5%]  (Warmup)
## Chain 1 Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1 Iteration:  300 / 2000 [ 15%]  (Warmup)
## Chain 1 Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1 Iteration:  500 / 2000 [ 25%]  (Warmup)
## Chain 1 Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1 Iteration:  700 / 2000 [ 35%]  (Warmup)
## Chain 1 Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1 Iteration:  900 / 2000 [ 45%]  (Warmup)
## Chain 1 Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1 Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1 Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 1 Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1 Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 1 Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1 Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 1 Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1 Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 1 Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1 Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 1 Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1 finished in 0.5 seconds.
## Chain 2 Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2 Iteration:  100 / 2000 [  5%]  (Warmup)
## Chain 2 Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2 Iteration:  300 / 2000 [ 15%]  (Warmup)
## Chain 2 Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2 Iteration:  500 / 2000 [ 25%]  (Warmup)
## Chain 2 Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2 Iteration:  700 / 2000 [ 35%]  (Warmup)
## Chain 2 Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2 Iteration:  900 / 2000 [ 45%]  (Warmup)
## Chain 2 Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2 Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2 Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 2 Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2 Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 2 Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2 Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 2 Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2 Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 2 Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2 Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 2 Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2 finished in 0.5 seconds.
##
## Both chains finished successfully.
## Mean chain execution time: 0.5 seconds.
## Total execution time: 1.2 seconds.
```

```r
ma_m0_pubpos <- update(
  ma_m0_all, newdata = subset(d1, PublishedPos == 1))
```

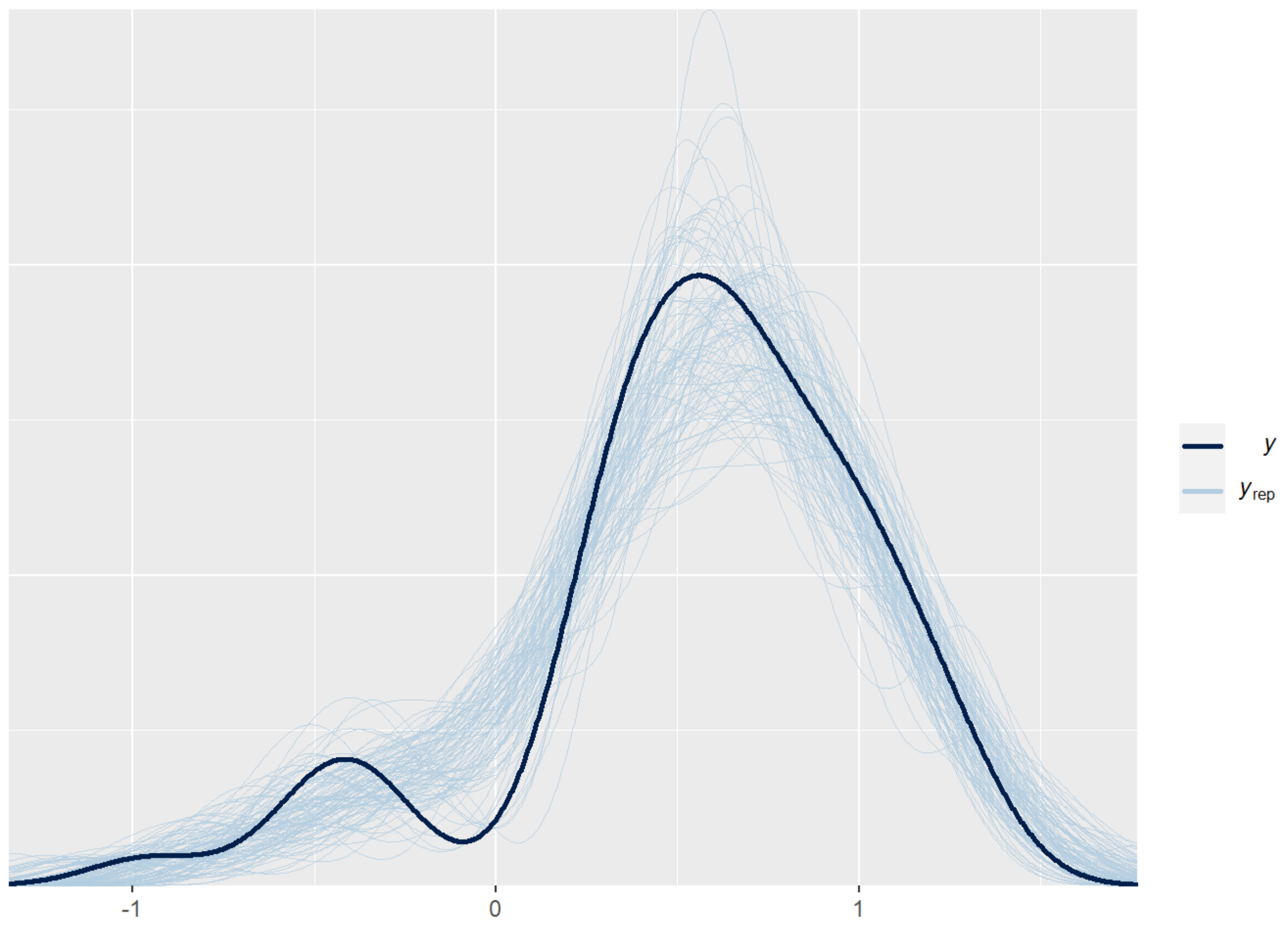
```
## Running MCMC with 2 sequential chains, with 2 thread(s) per chain...
## 
## Chain 1 Iteration:    1 / 2000 [  0%]  (Warmup) 
## Chain 1 Iteration:  100 / 2000 [  5%]  (Warmup) 
## Chain 1 Iteration:  200 / 2000 [ 10%]  (Warmup) 
## Chain 1 Iteration:  300 / 2000 [ 15%]  (Warmup) 
## Chain 1 Iteration:  400 / 2000 [ 20%]  (Warmup) 
## Chain 1 Iteration:  500 / 2000 [ 25%]  (Warmup) 
## Chain 1 Iteration:  600 / 2000 [ 30%]  (Warmup) 
## Chain 1 Iteration:  700 / 2000 [ 35%]  (Warmup) 
## Chain 1 Iteration:  800 / 2000 [ 40%]  (Warmup) 
## Chain 1 Iteration:  900 / 2000 [ 45%]  (Warmup) 
## Chain 1 Iteration: 1000 / 2000 [ 50%]  (Warmup) 
## Chain 1 Iteration: 1001 / 2000 [ 50%]  (Sampling) 
## Chain 1 Iteration: 1100 / 2000 [ 55%]  (Sampling) 
## Chain 1 Iteration: 1200 / 2000 [ 60%]  (Sampling) 
## Chain 1 Iteration: 1300 / 2000 [ 65%]  (Sampling) 
## Chain 1 Iteration: 1400 / 2000 [ 70%]  (Sampling) 
## Chain 1 Iteration: 1500 / 2000 [ 75%]  (Sampling) 
## Chain 1 Iteration: 1600 / 2000 [ 80%]  (Sampling) 
## Chain 1 Iteration: 1700 / 2000 [ 85%]  (Sampling) 
## Chain 1 Iteration: 1800 / 2000 [ 90%]  (Sampling) 
## Chain 1 Iteration: 1900 / 2000 [ 95%]  (Sampling) 
## Chain 1 Iteration: 2000 / 2000 [100%]  (Sampling) 
## Chain 1 finished in 0.6 seconds.
## Chain 2 Iteration:    1 / 2000 [  0%]  (Warmup) 
## Chain 2 Iteration:  100 / 2000 [  5%]  (Warmup) 
## Chain 2 Iteration:  200 / 2000 [ 10%]  (Warmup) 
## Chain 2 Iteration:  300 / 2000 [ 15%]  (Warmup) 
## Chain 2 Iteration:  400 / 2000 [ 20%]  (Warmup) 
## Chain 2 Iteration:  500 / 2000 [ 25%]  (Warmup) 
## Chain 2 Iteration:  600 / 2000 [ 30%]  (Warmup) 
## Chain 2 Iteration:  700 / 2000 [ 35%]  (Warmup) 
## Chain 2 Iteration:  800 / 2000 [ 40%]  (Warmup) 
## Chain 2 Iteration:  900 / 2000 [ 45%]  (Warmup) 
## Chain 2 Iteration: 1000 / 2000 [ 50%]  (Warmup) 
## Chain 2 Iteration: 1001 / 2000 [ 50%]  (Sampling) 
## Chain 2 Iteration: 1100 / 2000 [ 55%]  (Sampling) 
## Chain 2 Iteration: 1200 / 2000 [ 60%]  (Sampling) 
## Chain 2 Iteration: 1300 / 2000 [ 65%]  (Sampling) 
## Chain 2 Iteration: 1400 / 2000 [ 70%]  (Sampling) 
## Chain 2 Iteration: 1500 / 2000 [ 75%]  (Sampling) 
## Chain 2 Iteration: 1600 / 2000 [ 80%]  (Sampling) 
## Chain 2 Iteration: 1700 / 2000 [ 85%]  (Sampling) 
## Chain 2 Iteration: 1800 / 2000 [ 90%]  (Sampling) 
## Chain 2 Iteration: 1900 / 2000 [ 95%]  (Sampling) 
## Chain 2 Iteration: 2000 / 2000 [100%]  (Sampling) 
## Chain 2 finished in 0.5 seconds.
## 
## Both chains finished successfully.
## Mean chain execution time: 0.6 seconds.
## Total execution time: 1.3 seconds.
```

PPChecks:

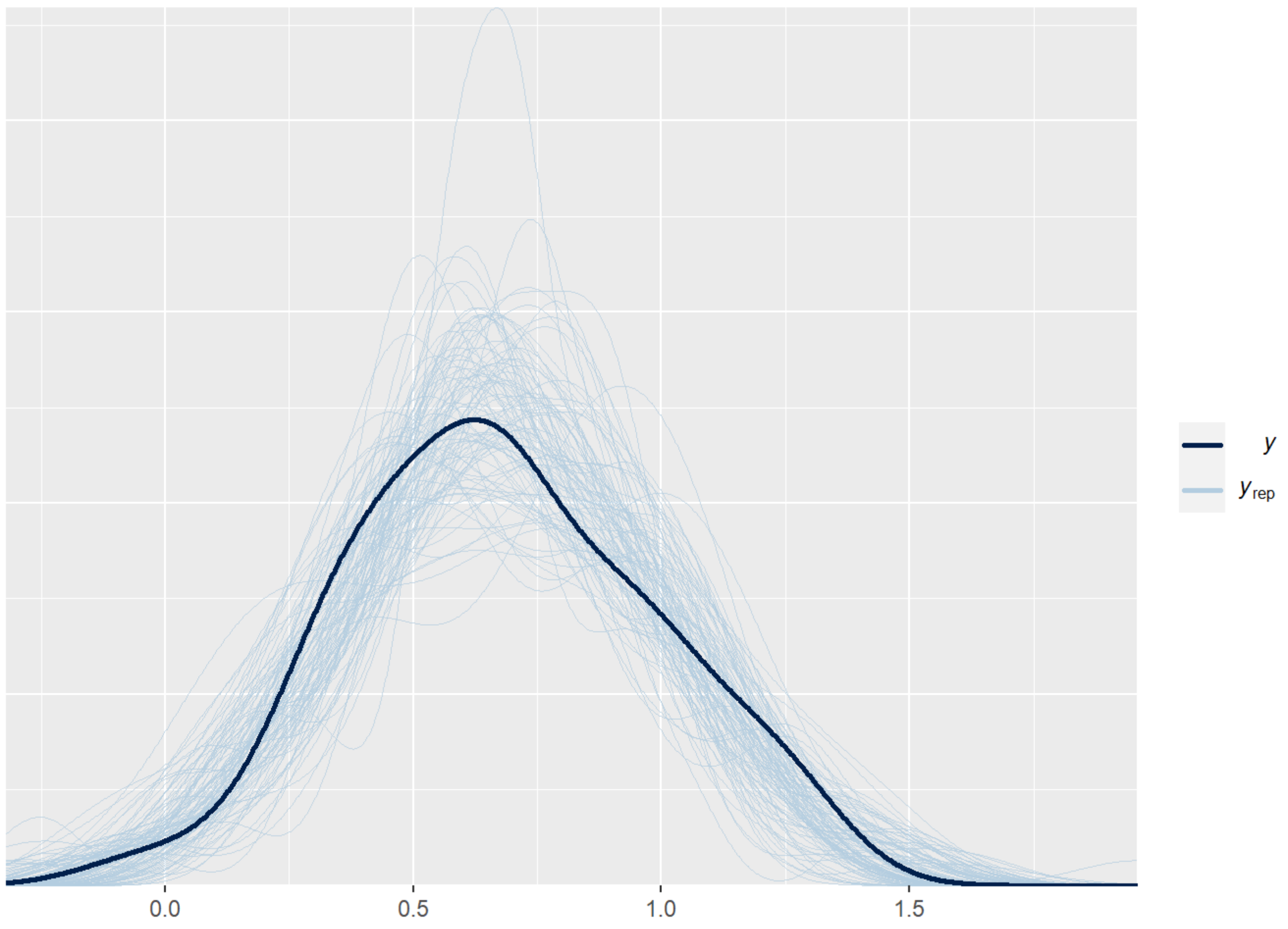
```
#pp checks of all the models
#all studies
pp_check(
  ma_m0_all, ndraws = 100)
```



```
#symmetric pubplication bias
pp_check(
  ma_m0_pub, ndraws = 100)
```

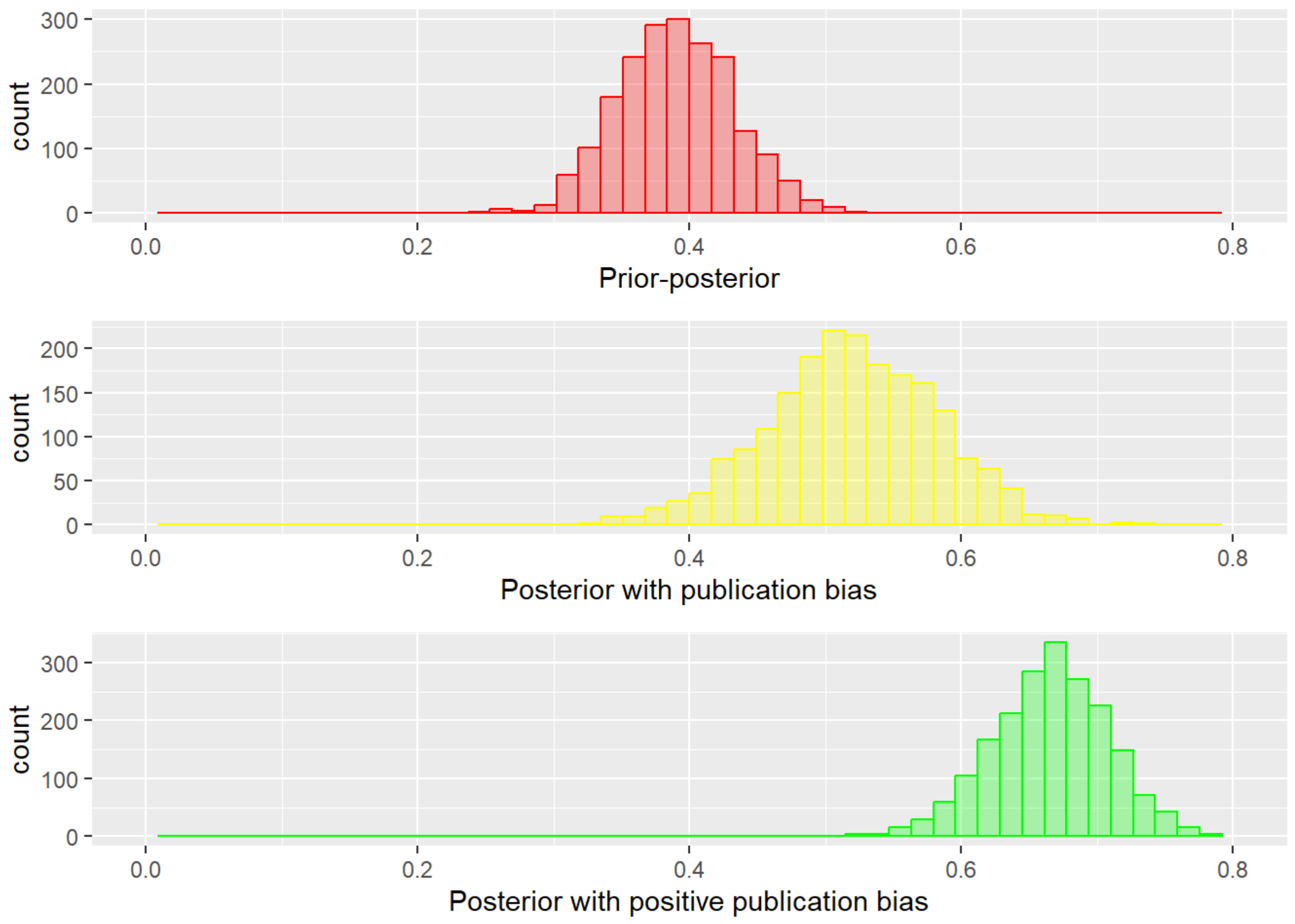


```
#symmetric positive pubblication bias
pp_check(
  ma_m0_pubpos, ndraws = 100)
```



## Visualization (TS)

```
posterior_prior <- as_draws_df(ma_m0_all)
posterior_pub <- as_draws_df(ma_m0_pub)
posterior_pubpos <- as_draws_df(ma_m0_pubpos)

p1 <- ggplot(posterior_prior) + geom_histogram(aes(b_Intercept), fill = "red", color = "red",
alpha = 0.3, bins = 50) + xlab("Prior-posterior") + xlim(0,0.8)

p2 <- ggplot(posterior_pub) + geom_histogram(aes(b_Intercept), fill = "yellow", color = "yell
ow", alpha = 0.3, bins = 50) + xlab("Posterior with publication bias")+ xlim(0,0.8)

p3 <- ggplot(posterior_pubpos) + geom_histogram(aes(b_Intercept), fill = "green", color = "gr
een", alpha = 0.3, bins = 50)+ xlab("Posterior with positive publication bias")+ xlim(0,0.8)

gridExtra::grid.arrange(p1, p2, p3)
```



The plot shows that including a publication bias or a positive publication bias predicts a very different estimate of the effect size. The models affected by a publication bias expect a higher effect size than the model not affected by the publication bias. Because of this, we could expect that a meta analysis would not represent the true population level and therefore we should always consider a possible publication bias.

Question 2

2. What is the current evidence for distinctive vocal patterns in schizophrenia? Use the data from Parola et al (2020) - https://www.dropbox.com/s/0l9ur0gaabr80a8/Matrix_MetaAnalysis_Diagnosis_updated290719.xlsx?dl=0 (https://www.dropbox.com/s/0l9ur0gaabr80a8/Matrix_MetaAnalysis_Diagnosis_updated290719.xlsx?dl=0) - focusing on pitch variability (PITCH_F0SD). Describe the data available (studies, participants). Using the model from question 1 analyze the data, visualize and report the findings: population level effect size; how well studies reflect it; influential studies, publication bias. BONUS question: assess the effect of task on the estimates (model comparison with baseline model)

Reshaping data frame (EL & LL)

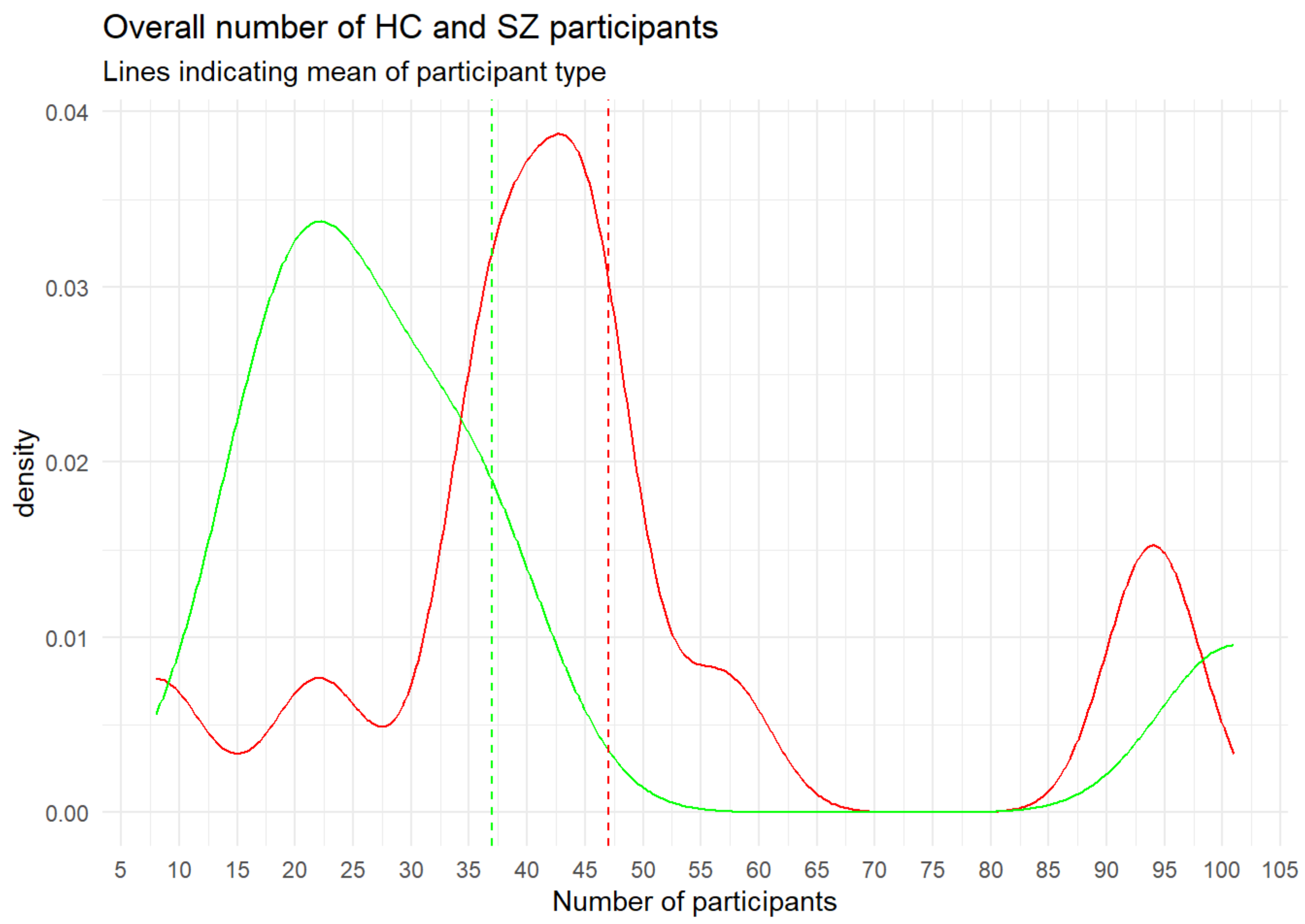
```
d_real <- readxl::read_excel("Matrix_MetaAnalysis.xlsx")
```

```
d_real_filtered <- d_real %>% dplyr::select(Title:FEMALE_HC,PITCH_F0SD_HC_M:PITCH_F0SD_SZ_SD)
%>%
  mutate("SD_POOLED" = sqrt((PITCH_F0SD_HC_SD^2 + PITCH_F0SD_SZ_SD^2)/2)) %>% #Creating SD_po
oled
  mutate("EFFECT_SIZE" = (PITCH_F0SD_SZ_M-PITCH_F0SD_HC_M)/SD_POOLED) %>%  #Creating Effect_s
ize collumn
  filter(is.na(EFFECT_SIZE) != TRUE & MALE_SZ != "NR") #Filter NA away
```

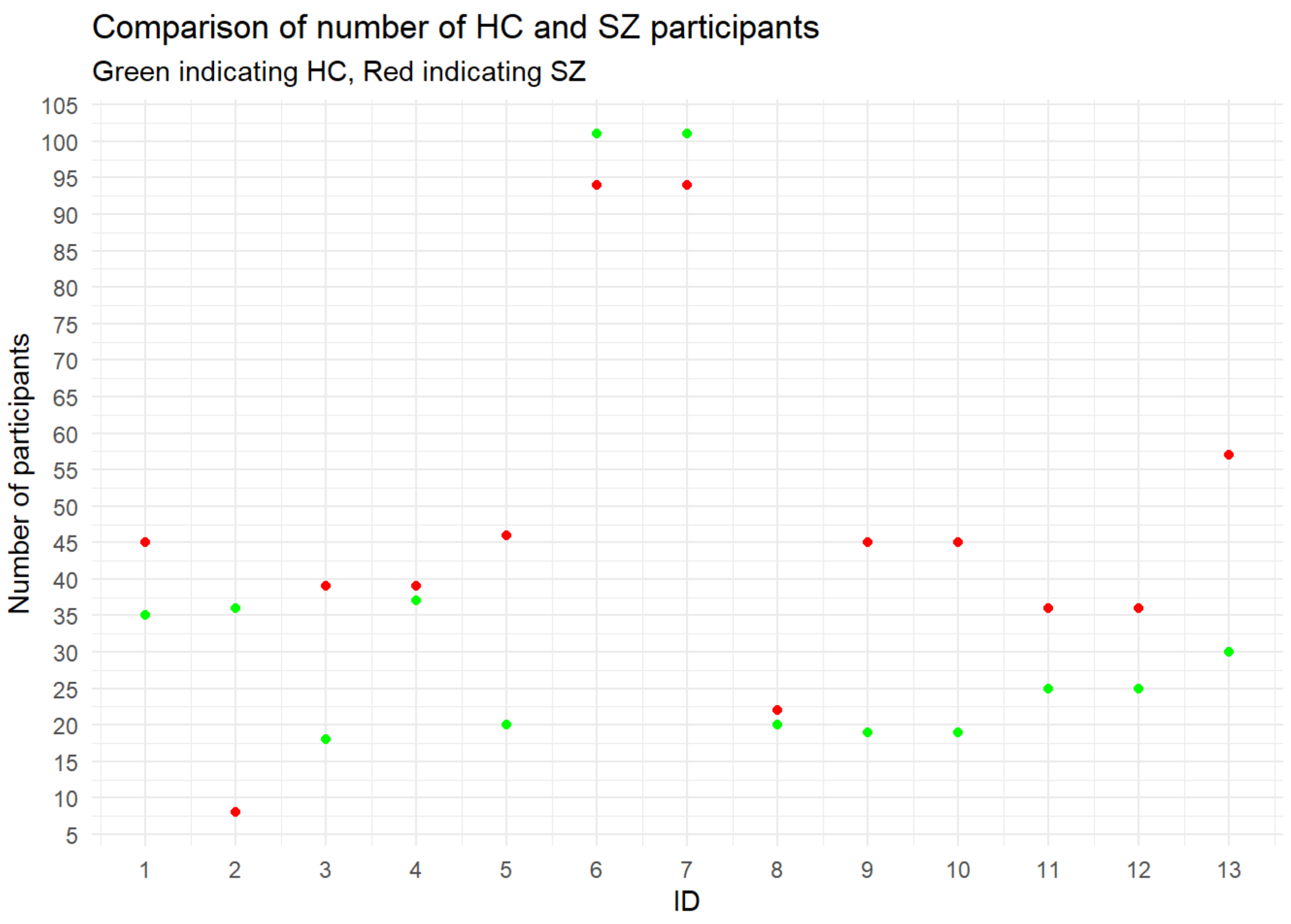
```
#Creating column with effect error
d_real_filtered[6:9] <- sapply(d_real_filtered[6:9], as.numeric)

d_real_filtered <- d_real_filtered %>%
  mutate("Pitch_SD" = (PITCH_F0SD_HC_SD+PITCH_F0SD_SZ_SD)/2, "N_participants" = MALE_SZ + FEM
ALE_SZ + MALE_HC + FEMALE_HC)
```

Visual investigation (NV)

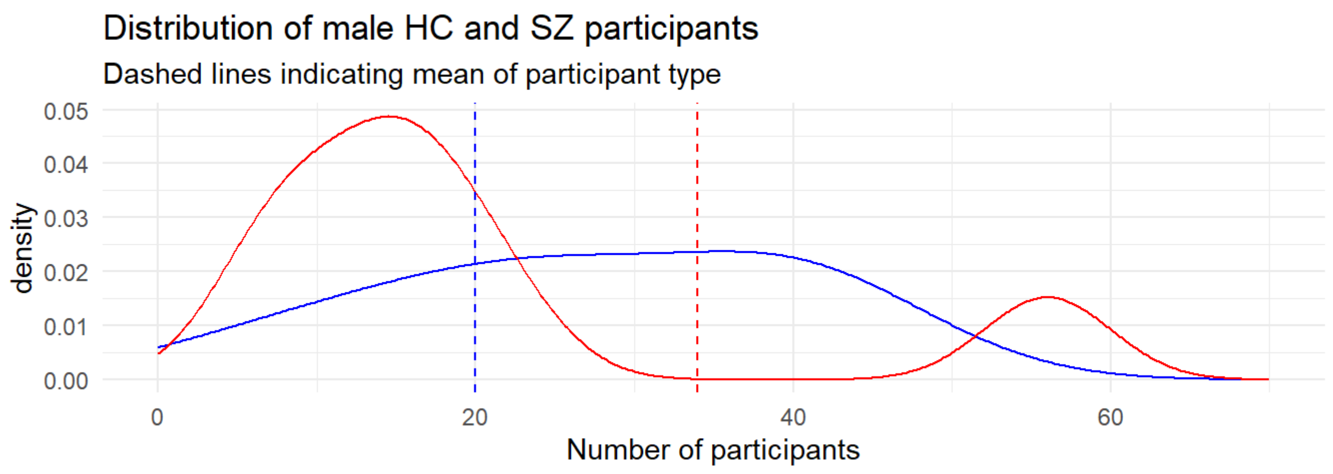
```
#transforming the participant coloumns in order to plot them
d_real_graph <- d_real_filtered[-4,]

d_real_graph <- d_real_graph %>%
  mutate(
    n_participants_sz = FEMALE_SZ + MALE_SZ,
    n_participants_hc = FEMALE_HC + MALE_HC,
    sz_mean = round(mean(n_participants_sz),0),
    hc_mean =round(mean(n_participants_hc),0),
    hc_male_mean =round(mean(MALE_HC),0),
    sz_male_mean =round(mean(MALE_SZ),0),
    hc_female_mean =round(mean(FEMALE_HC),0),
    sz_female_mean =round(mean(FEMALE_SZ),0),
  ) %>%
  rowid_to_column("ID")

#plotting the number of participants per study for the HC and SZ
d_real_graph %>%
  ggplot() +
  geom_density(aes(n_participants_sz), alpha = .2, color = "red") +
  geom_density(aes(n_participants_hc), alpha = .2, color = "green") +
  geom_vline(xintercept = d_real_graph$sz_mean, linetype = "dashed", color = "red") +
  geom_vline(xintercept = d_real_graph$hc_mean, linetype = "dashed", color = "green") +
  labs(title = "Overall number of HC and SZ participants", subtitle = "Lines indicating mean
of participant type") +
  xlab("Number of participants") +
  scale_x_continuous(breaks = scales::pretty_breaks(n = 15)) +
  theme_minimal()+ scale_fill_manual(name = "Participant type", values = c(SZ = "red", HC =
"green"))
```

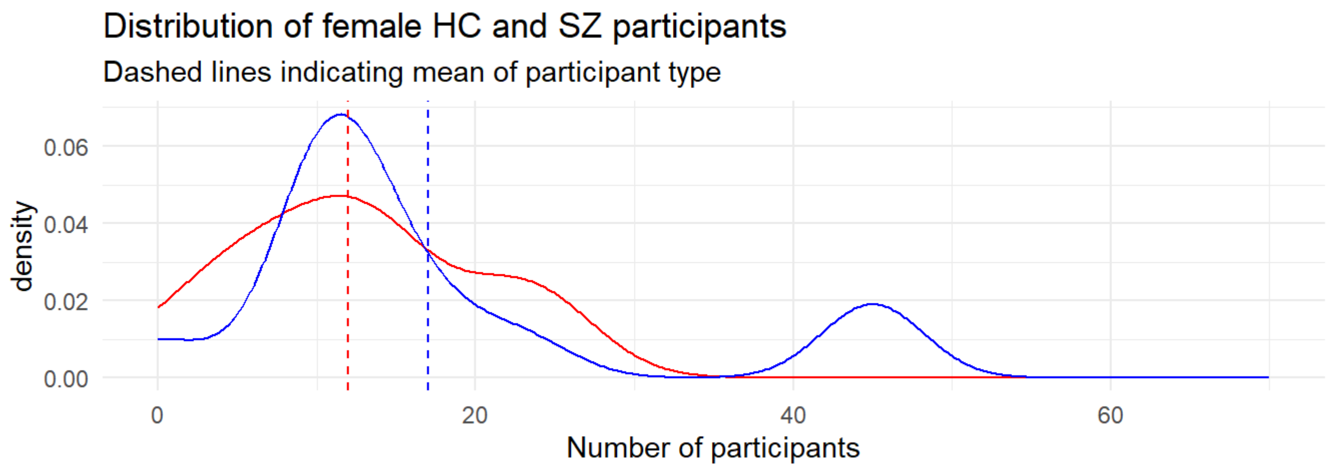


```
d_real_graph %>%
  ggplot(aes(x = ID)) +
  geom_point(aes(y =n_participants_sz), color = "red") +
  geom_point(aes(y =n_participants_hc), color = "green") +
  labs(title = "Comparison of number of HC and SZ participants", subtitle = "Green indicating
HC, Red indicating SZ") +
  ylab("Number of participants") +
  scale_y_continuous(breaks = scales::pretty_breaks(n = 15))+
  scale_x_continuous(breaks = scales::pretty_breaks(n = 13))+ theme_minimal() + scale_fill_ma
nual(name = "Participant type", values = c(SZ = "red", HC = "green"))
```


Comparison of number of HC and SZ participants
Green indicating HC, Red indicating SZ
Number of participants
105
100
95
90
85
80
75
70
65
60
55
50
45
40
35
30
25
20
15
10
5
1
2
3
4
5
6
7
8
9
10
11
12
13
ID

```
plot_female <- d_real_graph %>%
  ggplot() +
  xlim(0, 70)+
  geom_density(aes(FEMALE_SZ), alpha = .2, color = "red")+
  geom_density(aes(FEMALE_HC), alpha = .2, color = "blue") +
  geom_vline(xintercept = d_real_graph$hc_female_mean, linetype = "dashed", color = "blue") +
  geom_vline(xintercept = d_real_graph$sz_female_mean, color = "red", linetype = "dashed") +
  labs(title = "Distribution of female HC and SZ participants", subtitle = "Dashed lines indi
cating mean of participant type") +
  xlab("Number of participants") +
  theme_minimal()+ scale_fill_manual(name = "Participant type", values = c(SZ = "red", HC =
"blue"))

plot_male <- d_real_graph %>%
  ggplot() +
  xlim(0, 70)+
  geom_density(aes(MALE_SZ), alpha = .2, color = "blue") +
  geom_density(aes(MALE_HC), alpha = .2, color = "red") +
  geom_vline(xintercept = d_real_graph$hc_male_mean, color = "blue", linetype = "dashed") +
  geom_vline(xintercept = d_real_graph$sz_male_mean, color = "red", linetype =
               "dashed")+
labs(title = "Distribution of male HC and SZ participants", subtitle = "Dashed lines indicati
ng mean of participant type") +
  xlab("Number of participants") +
  theme_minimal()+ scale_fill_manual(name = "Participant type", values = c(SZ = "red", HC =
"blue"))

gridExtra::grid.arrange(plot_male, plot_female)
```





In general, there is more male participants than female. There is also quite a gap in the number of SZ patients between the genders. There is a mean of 12 female SZ participants, and a mean of 34 male SZ participants.

Participant description (LL)

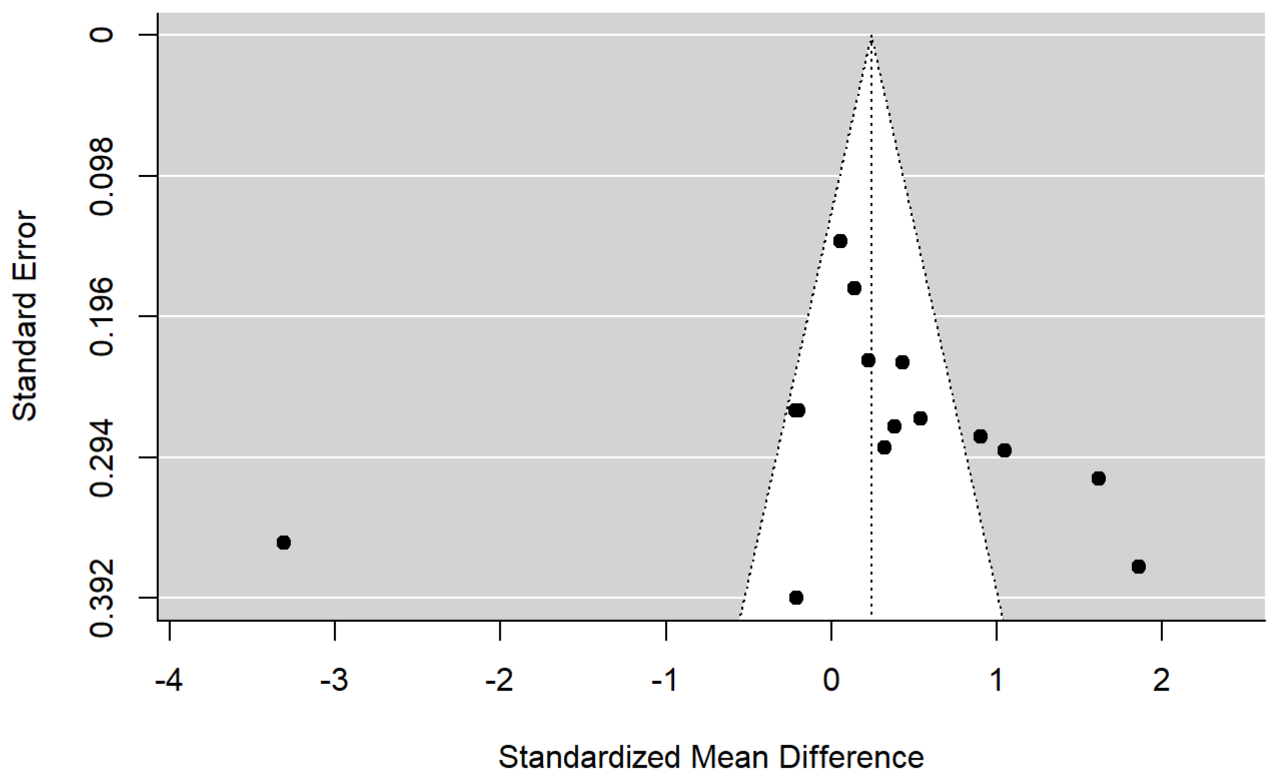
```
d_real %>%
  summarise(n = length(StudyID),
           participants_m_sz = mean(SAMPLE_SIZE_SZ,na.rm=T),
           participants_m_hc = mean(SAMPLE_SIZE_HC,na.rm=T),
           n_pitch_sz = length(PITCH_F0SD_SZ_M)-(sum(is.na(PITCH_F0SD_SZ_M))),
           Pitch_sz_mean=mean(PITCH_F0SD_SZ_M,na.rm=T),
           Pitch_sz_sd=mean(PITCH_F0SD_SZ_SD,na.rm=T),
            n_pitch_hc = length(PITCH_F0SD_HC_M)-(sum(is.na(PITCH_F0SD_HC_M))),
           Pitch_hc_mean=mean(PITCH_F0SD_HC_M,na.rm=T),
           Pitch_hc_sd=mean(PITCH_F0SD_HC_SD,na.rm=T))
```

```
## # A tibble: 1 × 9
##       n participants_m…¹ parti…² n_pit…³ Pitch…⁴ Pitch…⁵ n_pit…⁶ Pitch…⁷ Pitch…⁸
##   <int>            <dbl>   <dbl>   <int>   <dbl>   <dbl>   <int>   <dbl>   <dbl>
## 1    57             30.9    27.0      20    11.3    6.11      15    17.4    7.91
## # … with abbreviated variable names ¹participants_m_sz, ²participants_m_hc,
## #   ³n_pitch_sz, ⁴Pitch_sz_mean, ⁵Pitch_sz_sd, ⁶n_pitch_hc, ⁷Pitch_hc_mean,
## #   ⁸Pitch_hc_sd
```

Creating funnel plot to compare effect sizes from

different studies (TS)

```
PitchVariables<-escalc("SMD",
                  n1i=SAMPLE_SIZE_HC, n2i=SAMPLE_SIZE_SZ,
                  m1i=PITCH_F0SD_HC_M,
                  m2i=PITCH_F0SD_SZ_M,
                  sd1i=PITCH_F0SD_HC_SD,
                  sd2i=PITCH_F0SD_SZ_SD,
                  data = d_real)

Results <- rma(yi, vi, data=PitchVariables) # Performing meta-analysis - yi is effect size an
d vi is the effect size error
funnel(Results)
```



```
Results # print results
```

```
##
## Random-Effects Model (k = 15; tau^2 estimator: REML)
##
## tau^2 (estimated amount of total heterogeneity): 1.1965 (SE = 0.4817)
## tau (square root of estimated tau^2 value):      1.0938
## I^2 (total heterogeneity / total variability):   95.01%
## H^2 (total variability / sampling variability):  20.05
##
## Test for Heterogeneity:
## Q(df = 14) = 165.0993, p-val < .0001
##
## Model Results:
##
## estimate      se    zval    pval    ci.lb   ci.ub
##   0.2425  0.2915  0.8320  0.4054  -0.3288  0.8139
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This plot indicates that most effect sizes are above 0 with a relatively small standard error. However, there is one big outlier with an effect size of minus 3 and 5 other studies with an effect size below 0

Modeling on empirical data (LL & EL)

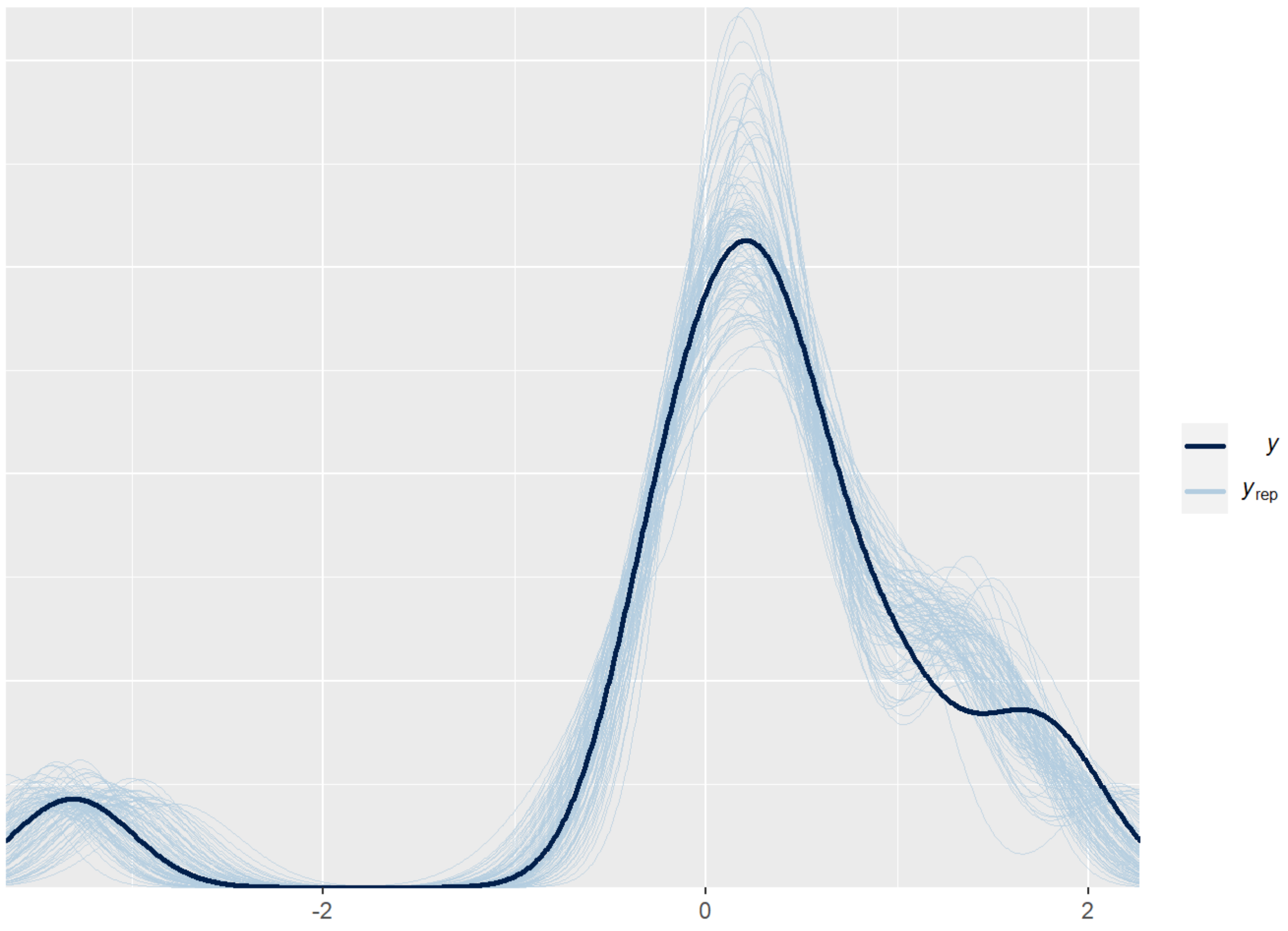
```
ma_f0_real <- bf(yi | se(vi) ~ 1 + (1 | Article))

get_prior(ma_f0_real,
          data = PitchVariables,
          family = gaussian)
```

```
##                   prior     class      coef   group resp dpar nlpar lb ub
##  student_t(3, 0.3, 2.5) Intercept
##    student_t(3, 0, 2.5)        sd                                  0
##    student_t(3, 0, 2.5)        sd           Article                0
##    student_t(3, 0, 2.5)        sd Intercept Article                0
##        source
##       default
##       default
##  (vectorized)
##  (vectorized)
```

```
ma_real <- brm(
  ma_f0_real,
  data = PitchVariables,
  save_pars = save_pars(all = T),
  family = gaussian,
  prior = ma_p0,
  sample_prior = T,
  backend = "cmdstanr",
  threads = threading(2),
  chains = 2,
  cores = 8,
  control = list(
    adapt_delta = 0.9,
    max_treedepth = 20
  ))
```

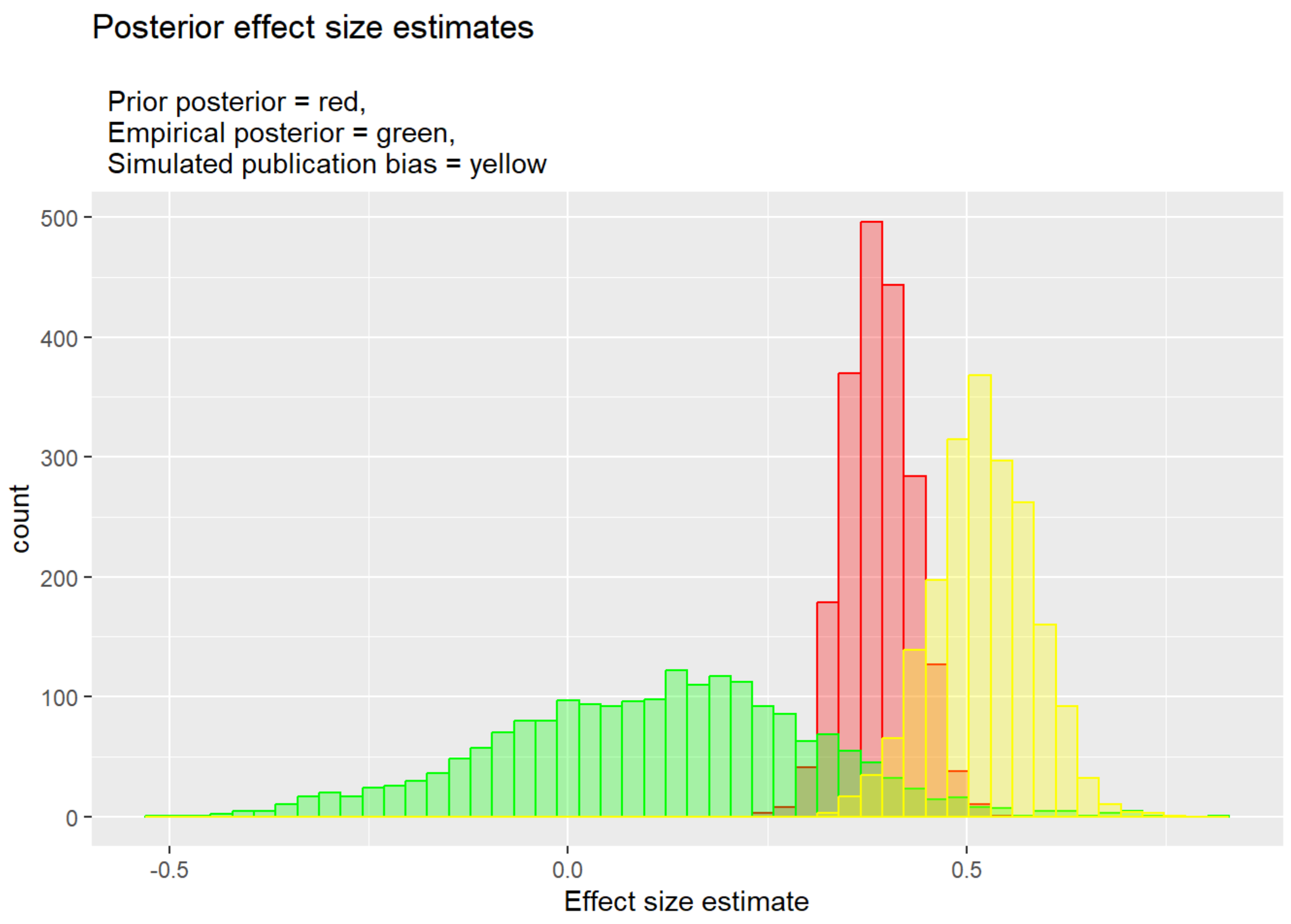
```
## Running MCMC with 2 chains, at most 8 in parallel, with 2 thread(s) per chain...
##
## Chain 1 Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1 Iteration:  100 / 2000 [  5%]  (Warmup)
## Chain 2 Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2 Iteration:  100 / 2000 [  5%]  (Warmup)
## Chain 1 Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1 Iteration:  300 / 2000 [ 15%]  (Warmup)
## Chain 1 Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1 Iteration:  500 / 2000 [ 25%]  (Warmup)
## Chain 1 Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2 Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2 Iteration:  300 / 2000 [ 15%]  (Warmup)
## Chain 2 Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2 Iteration:  500 / 2000 [ 25%]  (Warmup)
## Chain 2 Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2 Iteration:  700 / 2000 [ 35%]  (Warmup)
## Chain 1 Iteration:  700 / 2000 [ 35%]  (Warmup)
## Chain 1 Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2 Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2 Iteration:  900 / 2000 [ 45%]  (Warmup)
## Chain 2 Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2 Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1 Iteration:  900 / 2000 [ 45%]  (Warmup)
## Chain 1 Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1 Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1 Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 2 Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 2 Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2 Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 1 Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1 Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 2 Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2 Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 2 Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1 Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1 Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 1 Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2 Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 2 Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2 Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 1 Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 1 Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1 Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 2 Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2 finished in 0.7 seconds.
## Chain 1 Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1 finished in 0.9 seconds.
##
## Both chains finished successfully.
## Mean chain execution time: 0.8 seconds.
## Total execution time: 1.0 seconds.
```

```
pp_check(ma_real, ndraws=100)
```



Visualizing the findings (TS)

```
posterior_real <- as_draws_df(ma_real)

ggplot(NULL, aes(b_Intercept)) +
    geom_histogram(data = posterior_prior, fill = "red", col = "red",alpha = 0.3, bins = 50)
+
    geom_histogram(data = posterior_real, fill = "green", col = "green", alpha = 0.3, bins =
50)+
    geom_histogram(data = posterior_pub, fill = "yellow", col = "yellow", alpha = 0.3, bins =
50)+
  xlab("Effect size estimate") + labs(title = "Posterior effect size estimates",
                                      subtitle = "
  Prior posterior = red,
  Empirical posterior = green,
  Simulated publication bias = yellow")
```



```
summary(ma_real)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: yi | se(vi) ~ 1 + (1 | Article)
##    Data: PitchVariables (Number of observations: 15)
##   Draws: 2 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 2000
##
## Group-Level Effects:
## ~Article (Number of levels: 12)
##               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)     0.93      0.12     0.72     1.19 1.01      306      530
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     0.11      0.20    -0.30     0.48 1.02      207      401
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma     0.00      0.00     0.00     0.00   NA       NA       NA
##
## Draws were sampled using sample(hmc). 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).
```

These plots indicate, that when making a model that combines effect sizes from all of the different studies, we get a small population level effect size: Population level intercept = 0.13

This goes against our expectation that would be a higher positive effect size (that could be caused by a publication bias)