Assignment 2 - Meta-analysis of pitch in schizophrenia

Riccardo Fusaroli

16/8/2022

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 data set of *100 studies* with a *mean effect size of 0.4*, *average deviation by study of 0.4* and *measurement error of 0.8*.

N of participants should follow a *normal distribution* with *mean of 20*, *SD of 10*, but *no fewer than 10 participants)*.

The data should consist of: a) one row per study, with an *effect size mean* and *standard error*. Then: b) Build a proper Bayesian model to analyze the *simulated data*. c) 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), d) Discuss what this implies for your model. e) 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)

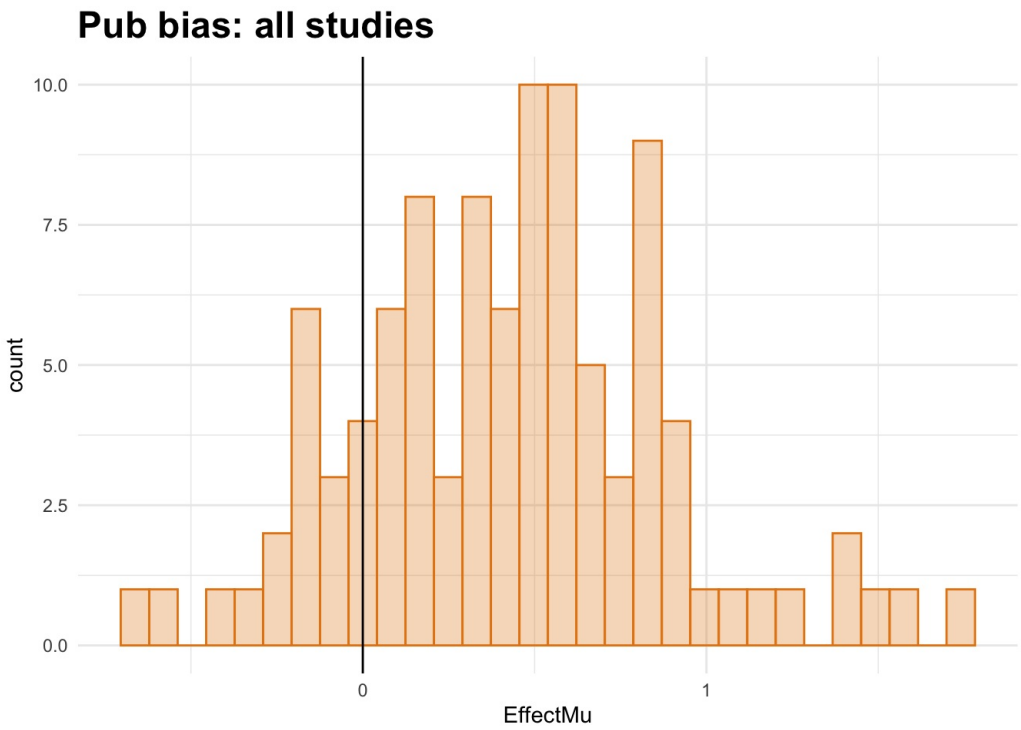
```
#Loading the packages
pacman::p_load(tidyverse, dplyr, tidybayes, ggplot2, ggridges, plyr, brms, cmdstanr, gridExtra, readxl)
```

Question 1

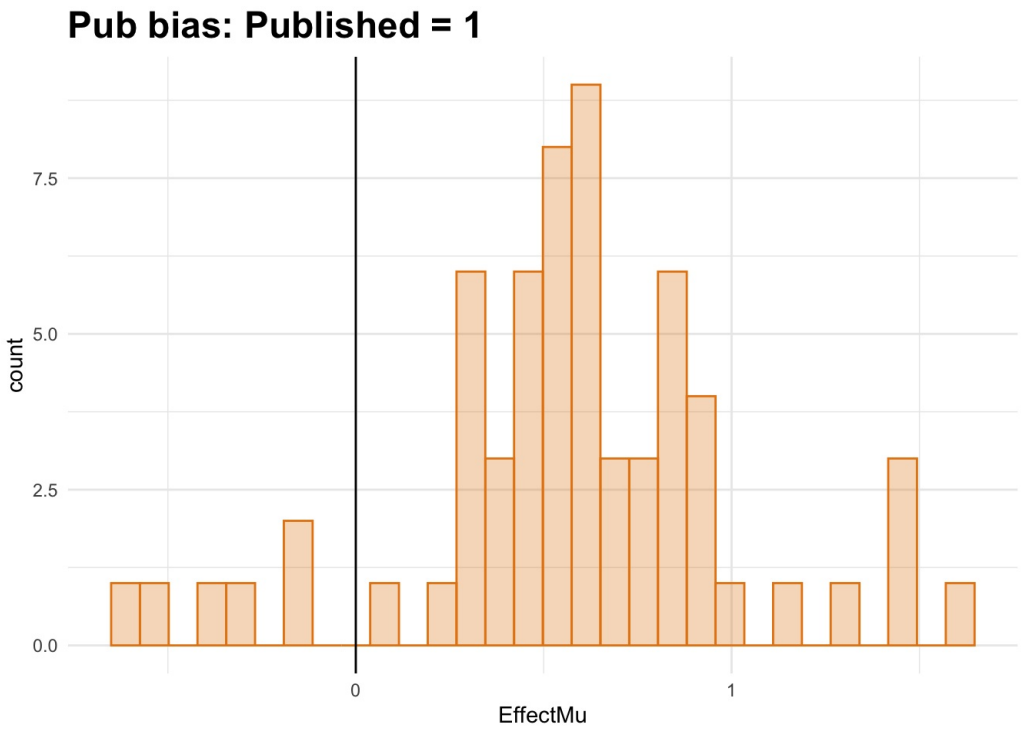
```
set.seed(578)

#Parameters for the simulation:
EffectMean <- 0.4                                                    #Effect between the populations.
Effect size in the simulation
StudySD <- 0.4                                                       #Difference between single studie
s. Average deviation by study
error <- 0.8                                                         #Error of measuring the pitch in
voice.

#Simulations
Studies <- 100                                                       #Simulating data set of 100 studi
es.

#How many participants each study will sample.
#Positive number of individuals per each study: defining the data frame
d <- tibble(
  Study = seq(Studies),
  Participants = round(msm::rtnorm(Studies, 20, 10, lower = 10), 0),
  EffectMu = NA, #effect size
  EffectSigma = NA, #uncertainty of the study
  StudyEffect = NA,
  Published = NA, #whether the study is published or not, will be simulated later
  PublishedPos = NA
)
```

Simulating the effect size, mean and standard error

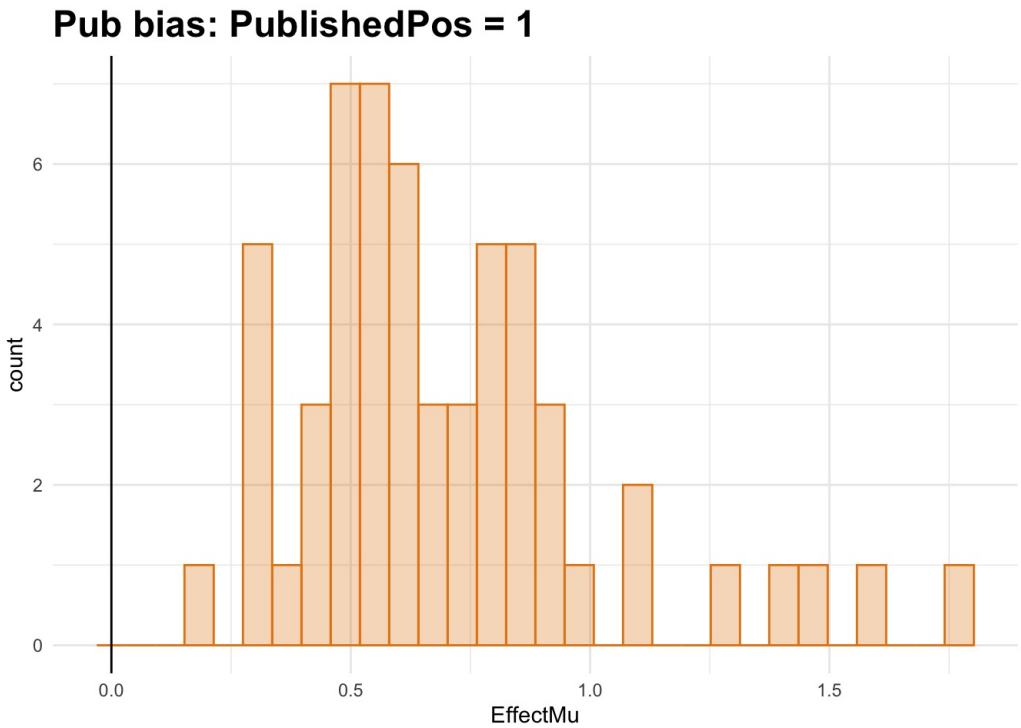
```
for (i in seq(Studies)) {
  d$StudyEffect[i] <- rnorm(1, EffectMean, StudySD)
  sampling <- rnorm(d$Participants[i], d$StudyEffect[i], error)
  d$EffectMu[i] <- mean(sampling)
  d$EffectSigma[i] <- sd(sampling)/sqrt(d$Participants[i]) #Standard error
  d$Published[i] <- ifelse(
    abs(d$EffectMu[i]) - (2*d$EffectSigma[i]) > 0, #If study is significant, it gets published by 90%
    rbinom(1, 1, .9), rbinom(1, 1, .1)) #if study is not significant, it gets published with prob. of 10%
  d$PublishedPos[i] <- ifelse(
    abs(d$EffectMu[i]) - (2*d$EffectSigma[i]) > 0 & d$EffectMu[i] > 0,  #
    rbinom(1, 1, .9), rbinom(1, 1, .1))
}

pub_bias_all <- ggplot(d) +
aes(x = EffectMu) +
geom_histogram(bins = 30L, fill = "#E68613", color = "#E68613", alpha = 0.3) +
labs(title = "Pub bias: all studies") +
geom_vline(xintercept = 0, color="black") +
theme_minimal() +
theme(plot.title = element_text(size = 18L, face = "bold"))

pub_bias_1 <- ggplot(subset(d, Published == 1)) +
aes(x = EffectMu) +
geom_histogram(bins = 30L, fill = "#E68613", color = "#E68613", alpha = 0.3) +
labs(title = "Pub bias: Published = 1") +
geom_vline(xintercept = 0, color="black") +
theme_minimal() +
theme(plot.title = element_text(size = 18L, face = "bold"))

pub_bias_pubpos_1 <- ggplot(subset(d, PublishedPos == 1)) +
aes(x = EffectMu) +
geom_histogram(bins = 30L, fill = "#E68613", color = "#E68613", alpha = 0.3) +
labs(title = "Pub bias: PublishedPos = 1") +
geom_vline(xintercept = 0, color="black") +
theme_minimal() +
theme(plot.title = element_text(size = 18L, face = "bold"))

pub_bias_all
```




```
pub_bias_1
```




```
pub_bias_pubpos_1
```



Meta-analytic multilevel modeling

```
f1 <- bf(EffectMu | se(EffectSigma) ~ 1 + (1 | Study))
get_prior(f1, d, gaussian)
```

Setting priors on the underlying level and not the data itself

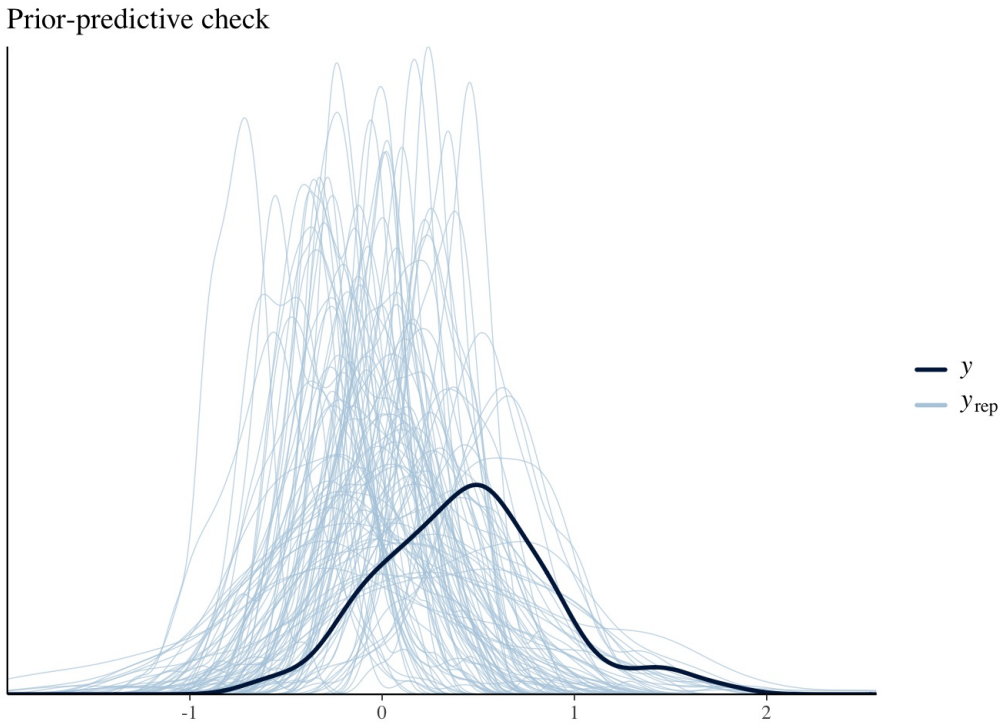
```
model_1_p <- c(
  prior(normal(0, 0.3), class = Intercept), #average undelying distribution of effects, any effect between -0.6 a
nd 0.6 is possible.
  prior(normal(0, 0.3), class = sd) #how much on average the studies will deviate from each other. Uncertainty of
the studies.
)
```

Plotting prior-predictive model

```
model_1_prior <- brm(
  f1,
  data = d,
  family = gaussian,
  prior = model_1_p,
  sample_prior = "only",
  backend = "cmdstanr",
  threads = threading(2),
  chains = 2,
  core = 2,
  control = list(adapt_delt = 0.99, max_treedepth = 20))
```

```
## Start sampling
```

```
pp_check(model_1_prior, ndraws=100) + labs(title = "Prior-predictive check")
```



Plotting posterior-predictive models #Differs every time when re-running, as I am not using “seed”, therefore looks different in the Word file.

```
#Fitting the models on full data set and the two publication biases
model_1_post <- brm(
  f1,
  data = d,
  family = gaussian,
  prior = model_1_p,
  sample_prior = T,
  backend = "cmdstanr",
  threads = threading(2),
  chains = 2,
  core = 2,
  control = list(adapt_delt = 0.99, max_treedepth = 20))
```

```
## Start sampling
```

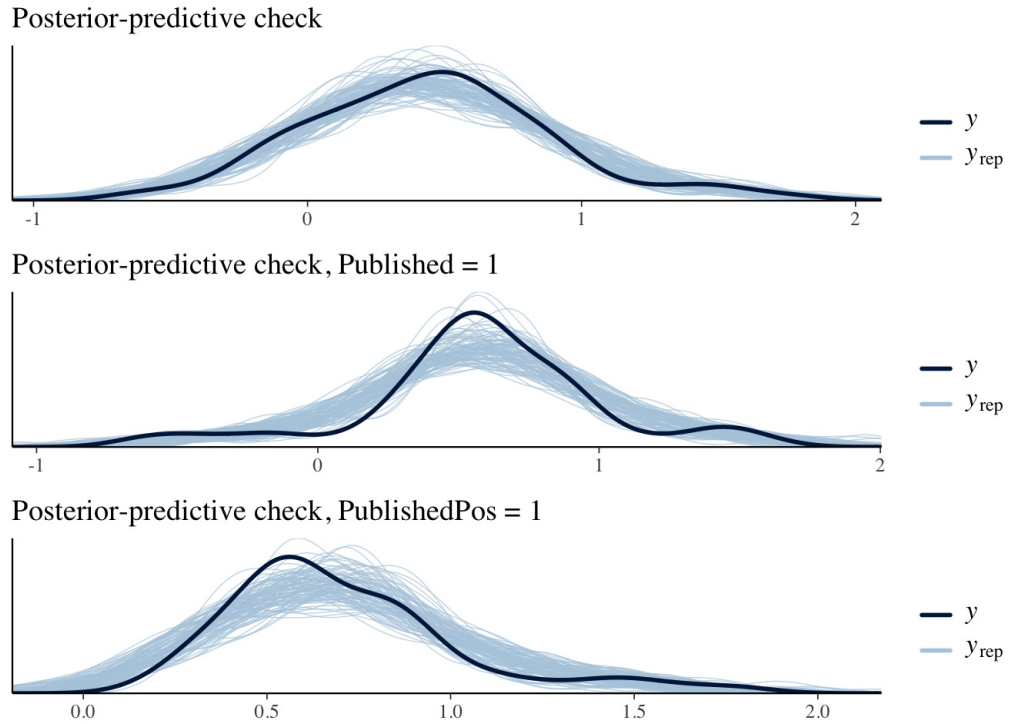
```
m1 <- pp_check(model_1_post, ndraws=100) + labs(title = "Posterior-predictive check")

model_1_post_pub <- update(model_1_post, newdata = subset(d, Published == 1))
```

```
## Start sampling
```

```
model_1_post_pubpos <- update(model_1_post, newdata = subset(d, PublishedPos == 1))
```

```
## Start sampling
```

```
m2 <- pp_check(model_1_post_pub, ndraws = 100) + labs(title = "Posterior-predictive check, Published = 1") #creat
ing plot
m3 <- pp_check(model_1_post_pubpos, ndraws = 100) + labs(title = "Posterior-predictive check, PublishedPos = 1")#
creating plot

grid.arrange(m1, m2, m3)
```



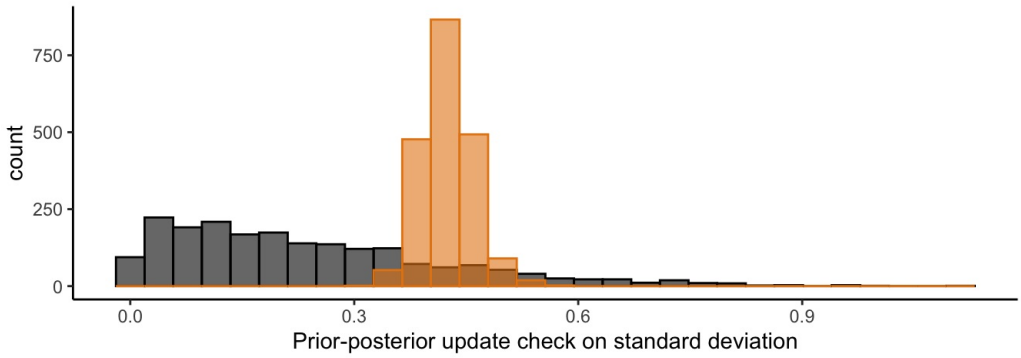
Plotting the estimates of these models

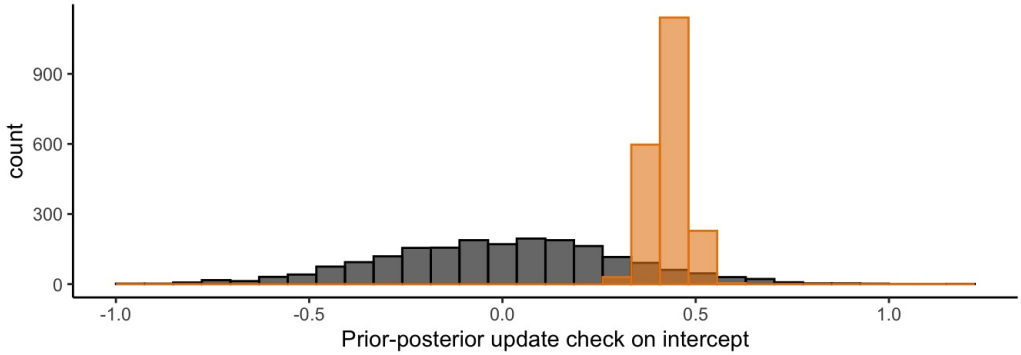
```
posterior_2 <- as_draws_df(model_1_post)
variables(posterior_2)
```

```
##   [1] "b_Intercept"            "sd_Study__Intercept"    "sigma"
##   [4] "r_Study[1,Intercept]"   "r_Study[2,Intercept]"   "r_Study[3,Intercept]"
##   [7] "r_Study[4,Intercept]"   "r_Study[5,Intercept]"   "r_Study[6,Intercept]"
##  [10] "r_Study[7,Intercept]"   "r_Study[8,Intercept]"   "r_Study[9,Intercept]"
##  [13] "r_Study[10,Intercept]"  "r_Study[11,Intercept]"  "r_Study[12,Intercept]"
##  [16] "r_Study[13,Intercept]"  "r_Study[14,Intercept]"  "r_Study[15,Intercept]"
##  [19] "r_Study[16,Intercept]"  "r_Study[17,Intercept]"  "r_Study[18,Intercept]"
##  [22] "r_Study[19,Intercept]"  "r_Study[20,Intercept]"  "r_Study[21,Intercept]"
##  [25] "r_Study[22,Intercept]"  "r_Study[23,Intercept]"  "r_Study[24,Intercept]"
##  [28] "r_Study[25,Intercept]"  "r_Study[26,Intercept]"  "r_Study[27,Intercept]"
##  [31] "r_Study[28,Intercept]"  "r_Study[29,Intercept]"  "r_Study[30,Intercept]"
##  [34] "r_Study[31,Intercept]"  "r_Study[32,Intercept]"  "r_Study[33,Intercept]"
##  [37] "r_Study[34,Intercept]"  "r_Study[35,Intercept]"  "r_Study[36,Intercept]"
##  [40] "r_Study[37,Intercept]"  "r_Study[38,Intercept]"  "r_Study[39,Intercept]"
##  [43] "r_Study[40,Intercept]"  "r_Study[41,Intercept]"  "r_Study[42,Intercept]"
##  [46] "r_Study[43,Intercept]"  "r_Study[44,Intercept]"  "r_Study[45,Intercept]"
##  [49] "r_Study[46,Intercept]"  "r_Study[47,Intercept]"  "r_Study[48,Intercept]"
##  [52] "r_Study[49,Intercept]"  "r_Study[50,Intercept]"  "r_Study[51,Intercept]"
##  [55] "r_Study[52,Intercept]"  "r_Study[53,Intercept]"  "r_Study[54,Intercept]"
##  [58] "r_Study[55,Intercept]"  "r_Study[56,Intercept]"  "r_Study[57,Intercept]"
##  [61] "r_Study[58,Intercept]"  "r_Study[59,Intercept]"  "r_Study[60,Intercept]"
##  [64] "r_Study[61,Intercept]"  "r_Study[62,Intercept]"  "r_Study[63,Intercept]"
##  [67] "r_Study[64,Intercept]"  "r_Study[65,Intercept]"  "r_Study[66,Intercept]"
##  [70] "r_Study[67,Intercept]"  "r_Study[68,Intercept]"  "r_Study[69,Intercept]"
##  [73] "r_Study[70,Intercept]"  "r_Study[71,Intercept]"  "r_Study[72,Intercept]"
##  [76] "r_Study[73,Intercept]"  "r_Study[74,Intercept]"  "r_Study[75,Intercept]"
##  [79] "r_Study[76,Intercept]"  "r_Study[77,Intercept]"  "r_Study[78,Intercept]"
##  [82] "r_Study[79,Intercept]"  "r_Study[80,Intercept]"  "r_Study[81,Intercept]"
##  [85] "r_Study[82,Intercept]"  "r_Study[83,Intercept]"  "r_Study[84,Intercept]"
##  [88] "r_Study[85,Intercept]"  "r_Study[86,Intercept]"  "r_Study[87,Intercept]"
##  [91] "r_Study[88,Intercept]"  "r_Study[89,Intercept]"  "r_Study[90,Intercept]"
##  [94] "r_Study[91,Intercept]"  "r_Study[92,Intercept]"  "r_Study[93,Intercept]"
##  [97] "r_Study[94,Intercept]"  "r_Study[95,Intercept]"  "r_Study[96,Intercept]"
## [100] "r_Study[97,Intercept]"  "r_Study[98,Intercept]"  "r_Study[99,Intercept]"
## [103] "r_Study[100,Intercept]" "prior_Intercept"        "prior_sd_Study"
## [106] "lprior"                 "lp__"
```

```
model_1_post_pub_draws <- as_draws_df(model_1_post_pub)

model_1_post_pubpos_draws <- as_draws_df(model_1_post_pubpos)

mod_1_df <- tibble(
  Model = "1",
  mean_est = mean(posterior_2$b_Intercept),
  upper = quantile(posterior_2$b_Intercept, 0.975),
  lower = quantile(posterior_2$b_Intercept, 0.025)
)

mod_2_df <- tibble(
  Model = "2",
  mean_est = mean(model_1_post_pub_draws$b_Intercept),
  upper = quantile(model_1_post_pub_draws$b_Intercept, 0.975),
  lower = quantile(model_1_post_pub_draws$b_Intercept, 0.025)
)

mod_3_df <- tibble(
  Model = "3",
  mean_est = mean(model_1_post_pubpos_draws$b_Intercept),
  upper = quantile(model_1_post_pubpos_draws$b_Intercept, 0.975),
  lower = quantile(model_1_post_pubpos_draws$b_Intercept, 0.025)
)

df_models_draws <- rbind(mod_1_df, mod_2_df, mod_3_df)


plots_est_p2 <- ggplot(df_models_draws) +
  geom_pointrange(aes(x= Model,y= mean_est,
                      ymin=lower,ymax=upper,
                      color = Model),alpha= 1) +
  xlab("Model") +
  ylab("Estimate")



plot1 <- ggplot(posterior_2) +
  geom_histogram(aes(prior_sd_Study), fill="black", color="black",alpha=0.6,) +
  geom_histogram(aes(sd_Study__Intercept), fill="#E68613", color="#E68613",alpha=0.6) + #Posterior
  theme_classic() +
   xlab("Prior-posterior update check on standard deviation")

plot2 <- ggplot(posterior_2) +
  geom_histogram(aes(prior_Intercept), fill="black", color="black",alpha=0.6,) +
  geom_histogram(aes(b_Intercept), fill="#E68613", color="#E68613",alpha=0.6) + #Posterior
  theme_classic() +
   xlab("Prior-posterior update check on intercept")
grid.arrange(plot1, plot2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```





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).

2.1) Describe the data available (studies, participants). 2.2) 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.

```
#Loading the data in:
ass_2_d <- read_excel("Matrix_MetaAnalysis_Diagnosis_updated290719.xlsx")
```

```
## New names:
## • `frequency` -> `frequency...68`
## • `frequency` -> `frequency...73`
## • `frequency` -> `frequency...78`
## • `frequency` -> `frequency...83`
## • `frequency` -> `frequency...88`
## • `frequency` -> `frequency...93`
## • `frequency` -> `frequency...98`
## • `variability` -> `variability...108`
## • `variability` -> `variability...113`
## • `variability` -> `variability...118`
## • `variability` -> `variability...123`
## • `variability` -> `variability...128`
```

Variables: - HC - healthy controls - F0 = Fundamental frequency (frequency of oscillation produced by tension of vocal folds during speech). Measured in seconds/Hz. Corresponds roughly to perceived pitch of speech.

Describe the data:

```
ass_2_d$StudyID <- as.factor(ass_2_d$StudyID)
length(levels(ass_2_d$StudyID)) #50 studies

#Changing variables from chr to num so I could compare them.
#For sample size:
ass_2_d$MALE_SZ <- as.numeric(ass_2_d$MALE_SZ)
ass_2_d$FEMALE_SZ <- as.numeric(ass_2_d$FEMALE_SZ)

ass_2_d$MALE_HC <- as.numeric(ass_2_d$MALE_HC)
ass_2_d$FEMALE_HC <- as.numeric(ass_2_d$FEMALE_HC)

#For age:
ass_2_d$AGE_M_SZ <- as.numeric(ass_2_d$AGE_M_SZ)
ass_2_d$AGE_SD_SZ <- as.numeric(ass_2_d$AGE_SD_SZ)

ass_2_d$AGE_M_HC <- as.numeric(ass_2_d$AGE_M_HC)
ass_2_d$AGE_SD_HC <- as.numeric(ass_2_d$AGE_SD_HC)
```

Data analysis: (Focus on pitch variability: PITCH_F0_HC_SD and PITCH_F0_SZ_SD for control and schizophrenic group.)

```
library(metafor)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loading required package: metadat
```

```
##
## Loading the 'metafor' package (version 3.8-1). For an
## introduction to the package please type: help(metafor)
```

```
#Converting existing outcomes of the studies to Cohen's D:
Outcome_ES <- escalc('SMD',
  n1i = SAMPLE_SIZE_SZ, n2i = SAMPLE_SIZE_HC,
  m1i = PITCH_F0_SZ_M, m2i=PITCH_F0_HC_M,
  sd1i = PITCH_F0_SZ_SD, sd2i=PITCH_F0_HC_SD,
  data = ass_2_d)
#head(Outcome_ES)
```

Yi - The EffectMu Vi - Standard error

Defining the formula for the model:

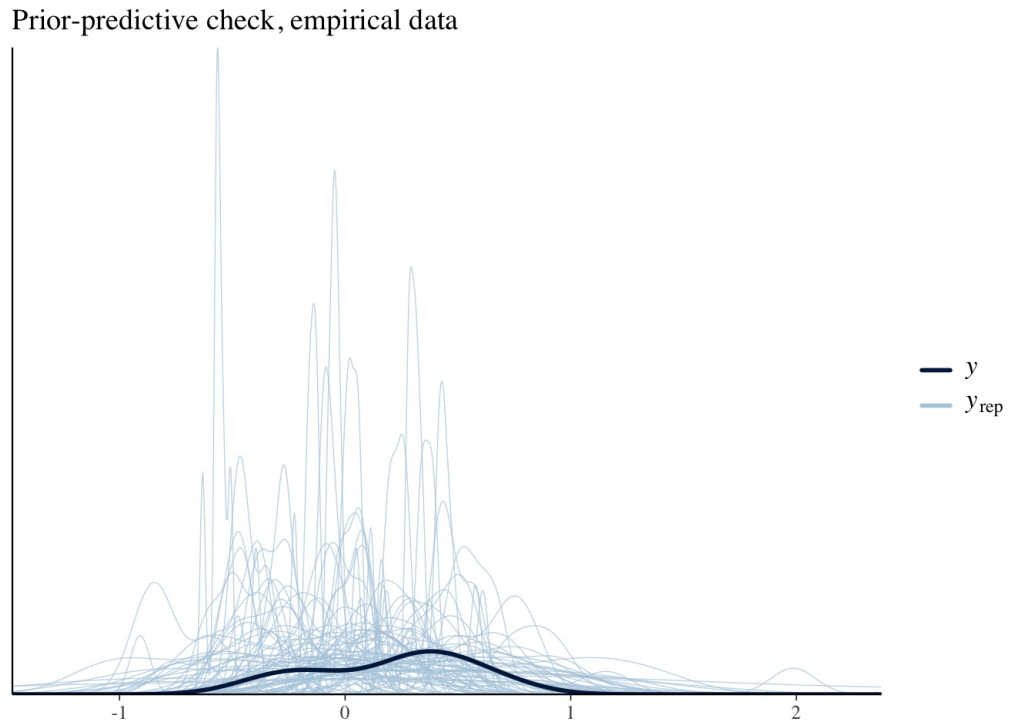
```
pitch_data_f <- bf(yi | se(vi) ~ 1 + (1 | StudyID))
#get_prior(pitch_data_f, data = Outcome_ES, gaussian)

#Same as in the model above
model_2_priors <- c(
  prior(normal(0, 0.3), class = Intercept),
  prior(normal(0, 0.3), class = sd)
)

model_p2_prior <- brm(
  pitch_data_f,
  data = Outcome_ES,
  family = gaussian,
  prior = model_2_priors,
  sample_prior = "only",
  backend = "cmdstanr",
  threads = threading(2),
  chains = 2,
  core = 2,
  control = list(adapt_delt = 0.99, max_treedepth = 20))
```

```
## Warning: Rows containing NAs were excluded from the model.
```

```
## Start sampling
```

```
## Running MCMC with 2 parallel 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 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 1 finished in 0.1 seconds.
## Chain 2 finished in 0.1 seconds.
##
## Both chains finished successfully.
## Mean chain execution time: 0.1 seconds.
## Total execution time: 0.2 seconds.
```

```
#update(model_p2_prior)
pp_check(model_p2_prior, ndraws=100) + labs(title = "Prior-predictive check, empirical data")
```



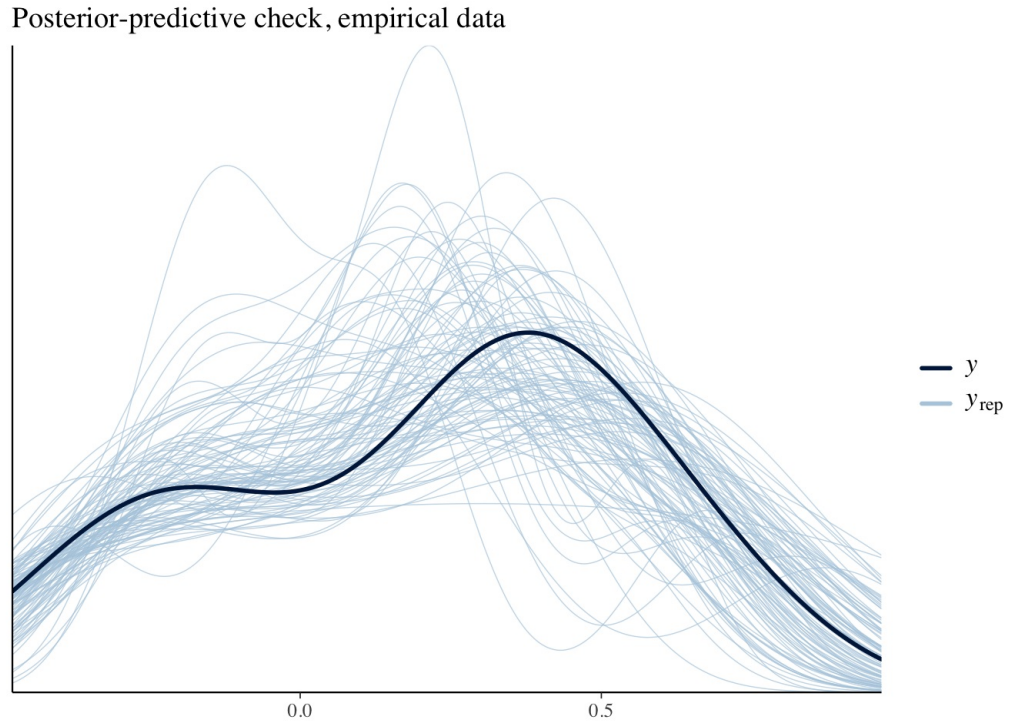

```
model_p2_post <- brm(
  pitch_data_f,
  data = Outcome_ES,
  family = gaussian,
  prior = model_2_priors,
  sample_prior = T,
  backend = "cmdstanr",
  threads = threading(2),
  chains = 2,
  core = 2,
  control = list(adapt_delt = 0.99, max_treedepth = 20))
```

```
## Warning: Rows containing NAs were excluded from the model.
```

```
## Start sampling
```

```
## Running MCMC with 2 parallel 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 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 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 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 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 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 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 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 1 Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 1 Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2 Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1 finished in 0.5 seconds.
## Chain 2 finished in 0.4 seconds.
##
## Both chains finished successfully.
## Mean chain execution time: 0.5 seconds.
## Total execution time: 0.6 seconds.
```

```
#update(model_p2_post)
pp_check(model_p2_post, ndraws=100) + labs(title = "Posterior-predictive check, empirical data")
```




```
#Outcome_ES
```

Posterior:

```
model_p2_draws <- as_draws_df(model_p2_post)
variables(model_p2_draws)
```

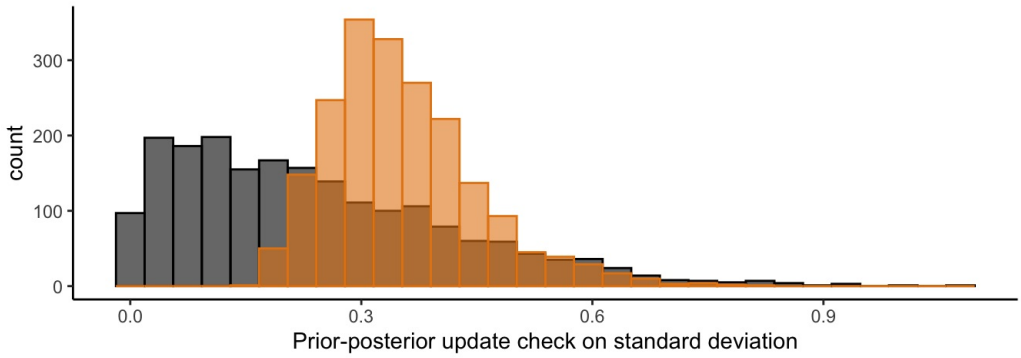
```
##  [1] "b_Intercept"             "sd_StudyID__Intercept"
##  [3] "sigma"                   "r_StudyID[1,Intercept]"
##  [5] "r_StudyID[5,Intercept]"  "r_StudyID[11,Intercept]"
##  [7] "r_StudyID[18,Intercept]" "r_StudyID[28,Intercept]"
##  [9] "r_StudyID[50,Intercept]" "prior_Intercept"
## [11] "prior_sd_StudyID"        "lprior"
## [13] "lp__"
```

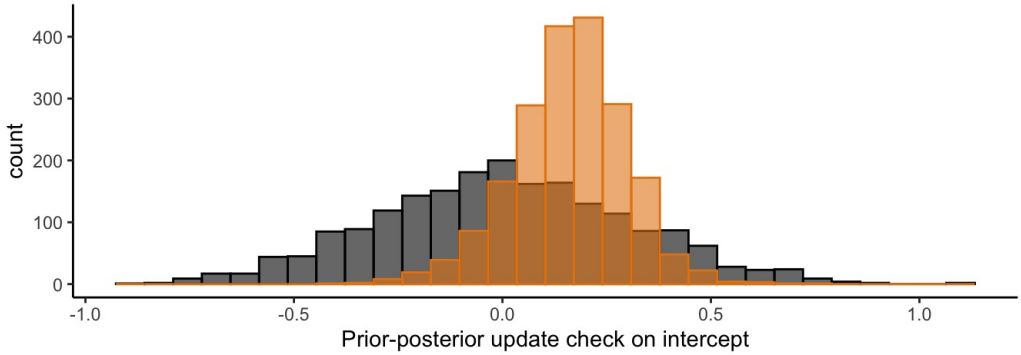
```
model_p2_draws
```

```
## # A draws_df: 1000 iterations, 2 chains, and 13 variables
##    b_Intercept sd_StudyID__Intercept sigma r_StudyID[1,Intercept]
## 1      0.09277                 0.37     0                  0.204
## 2     -0.02933                 0.36     0                  0.438
## 3      0.07495                 0.35     0                  0.442
## 4      0.02344                 0.31     0                  0.348
## 5      0.00068                 0.28     0                  0.370
## 6      0.22976                 0.39     0                  0.123
## 7      0.27548                 0.37     0                  0.078
## 8      0.23802                 0.27     0                  0.247
## 9      0.34855                 0.27     0                  0.111
## 10     0.09413                 0.22     0                  0.104
##    r_StudyID[5,Intercept] r_StudyID[11,Intercept] r_StudyID[18,Intercept]
## 1                   -0.46                   0.61                 1.5e-01
## 2                   -0.24                   0.65                 3.3e-01
## 3                   -0.39                   0.54                 1.6e-01
## 4                   -0.28                   0.57                 2.2e-01
## 5                   -0.37                   0.65                 2.6e-01
## 6                   -0.58                   0.41                -5.0e-02
## 7                   -0.56                   0.36                -4.6e-05
## 8                   -0.49                   0.48                 4.5e-02
## 9                   -0.79                   0.19                -9.3e-02
## 10                  -0.43                   0.38                 1.9e-01
##    r_StudyID[28,Intercept]
## 1                    0.242
## 2                    0.287
## 3                    0.462
## 4                    0.350
## 5                    0.173
## 6                    0.374
## 7                    0.131
## 8                   -0.076
## 9                    0.034
## 10                   0.116
## # ... with 1990 more draws, and 5 more variables
## # ... hidden reserved variables {'.chain', '.iteration', '.draw'}
```

```
plot1_emp <- ggplot(model_p2_draws) +
  geom_histogram(aes(prior_sd_StudyID), fill="black", color="black",alpha=0.6,) +
  geom_histogram(aes(sd_StudyID__Intercept), fill="#E68613", color="#E68613",alpha=0.6) + #Posterior
  theme_classic() +
   xlab("Prior-posterior update check on standard deviation")

plot2_emp <- ggplot(model_p2_draws) +
  geom_histogram(aes(prior_Intercept), fill="black", color="black",alpha=0.6,) +
  geom_histogram(aes(b_Intercept), fill="#E68613", color="#E68613",alpha=0.6) + #Posterior
  theme_classic() +
   xlab("Prior-posterior update check on intercept")

grid.arrange(plot1_emp, plot2_emp)
```

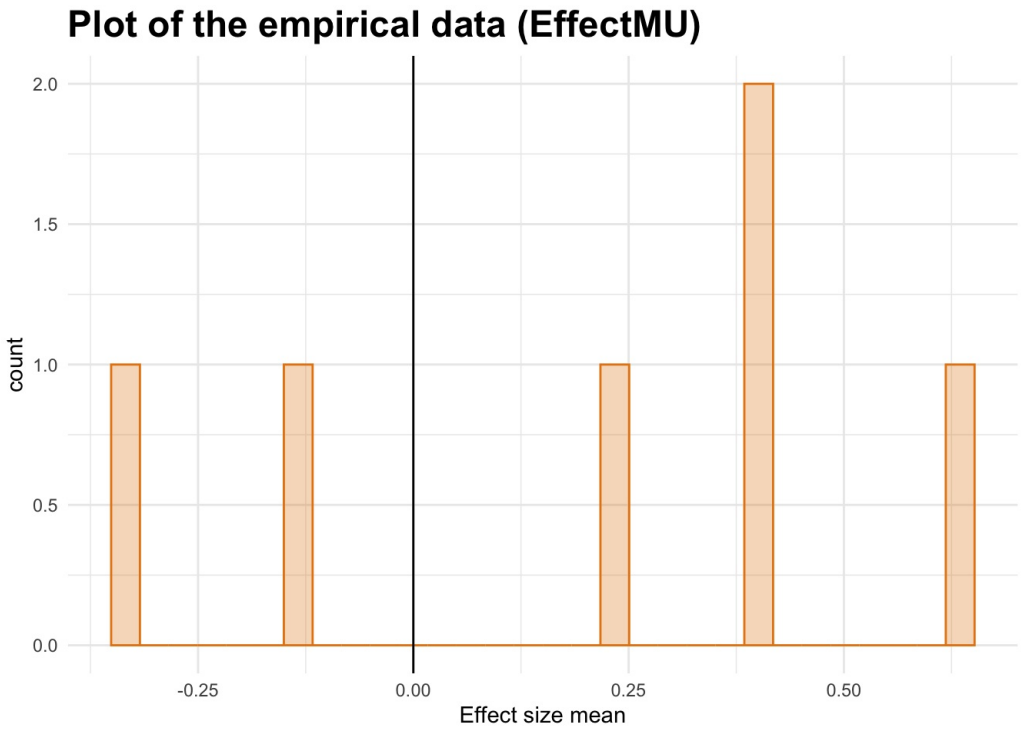
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```






```
pub_bias_real_d <- ggplot(Outcome_ES) +
aes(x = yi) +
geom_histogram(bins = 30L, fill = "#E68613", color = "#E68613", alpha = 0.3) +
labs(title = "Plot of the empirical data (EffectMU)") +
geom_vline(xintercept = 0, color="black") +
theme_minimal() +
theme(plot.title = element_text(size = 18L, face = "bold")) +
xlab("Effect size mean")
pub_bias_real_d
```

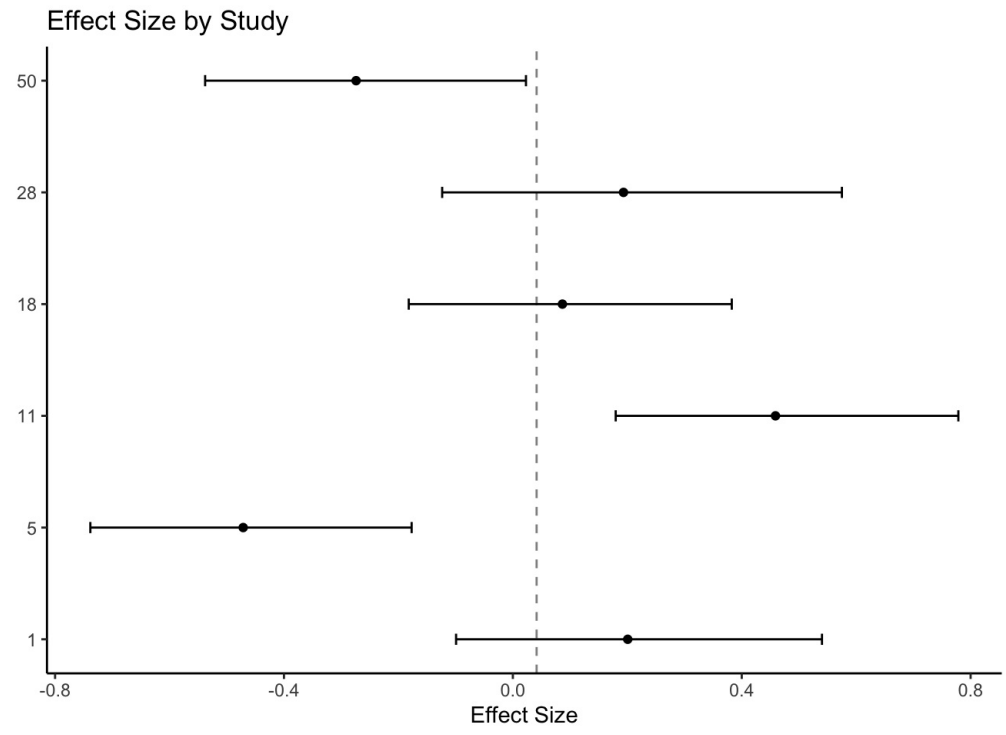
```
## Warning: Removed 51 rows containing non-finite values (`stat_bin()`).
```



Influential data points of the empirical

data:

```
#Creating a tibble to make it easier:
infl_studies <- tibble(
  Index = seq(1:6),
  StudyID = c(1, 5, 11, 18, 28, 50), #doing manually since its only 6 effect sizes.
  Mean = NA,
  Upper = NA,
  Lower = NA
)

#Adding the values:
infl_studies[1,3] <- mean(model_p2_draws$`r_StudyID[1,Intercept]`)
infl_studies[1,4] <- quantile(model_p2_draws$`r_StudyID[1,Intercept]`, 0.975)
infl_studies[1,5] <- quantile(model_p2_draws$`r_StudyID[1,Intercept]`, 0.025)

infl_studies[2,3] <- mean(model_p2_draws$`r_StudyID[5,Intercept]`)
infl_studies[2,4] <- quantile(model_p2_draws$`r_StudyID[5,Intercept]`, 0.975)
infl_studies[2,5] <- quantile(model_p2_draws$`r_StudyID[5,Intercept]`, 0.025)

infl_studies[3,3] <- mean(model_p2_draws$`r_StudyID[11,Intercept]`)
infl_studies[3,4] <- quantile(model_p2_draws$`r_StudyID[11,Intercept]`, 0.975)
infl_studies[3,5] <- quantile(model_p2_draws$`r_StudyID[11,Intercept]`, 0.025)

infl_studies[4,3] <- mean(model_p2_draws$`r_StudyID[18,Intercept]`)
infl_studies[4,4] <- quantile(model_p2_draws$`r_StudyID[18,Intercept]`, 0.975)
infl_studies[4,5] <- quantile(model_p2_draws$`r_StudyID[18,Intercept]`, 0.025)

infl_studies[5,3] <- mean(model_p2_draws$`r_StudyID[28,Intercept]`)
infl_studies[5,4] <- quantile(model_p2_draws$`r_StudyID[28,Intercept]`, 0.975)
infl_studies[5,5] <- quantile(model_p2_draws$`r_StudyID[28,Intercept]`, 0.025)

infl_studies[6,3] <- mean(model_p2_draws$`r_StudyID[50,Intercept]`)
infl_studies[6,4] <- quantile(model_p2_draws$`r_StudyID[50,Intercept]`, 0.975)
infl_studies[6,5] <- quantile(model_p2_draws$`r_StudyID[50,Intercept]`, 0.025)


#Plot of the studies
ggplot(data=infl_studies, aes(y=Index, x=Mean, xmin=Lower, xmax=Upper)) +
  geom_point() +
  geom_errorbarh(height=.1) +
  scale_y_continuous(name = "", breaks=1:nrow(infl_studies), labels=infl_studies$StudyID) +
  labs(title='Effect Size by Study', x='Effect Size', y = 'Study') +
  geom_vline(xintercept=0.0415, color='black', linetype='dashed', alpha=.5) +
  theme_classic()
```



Rerunning the model without influential studies:

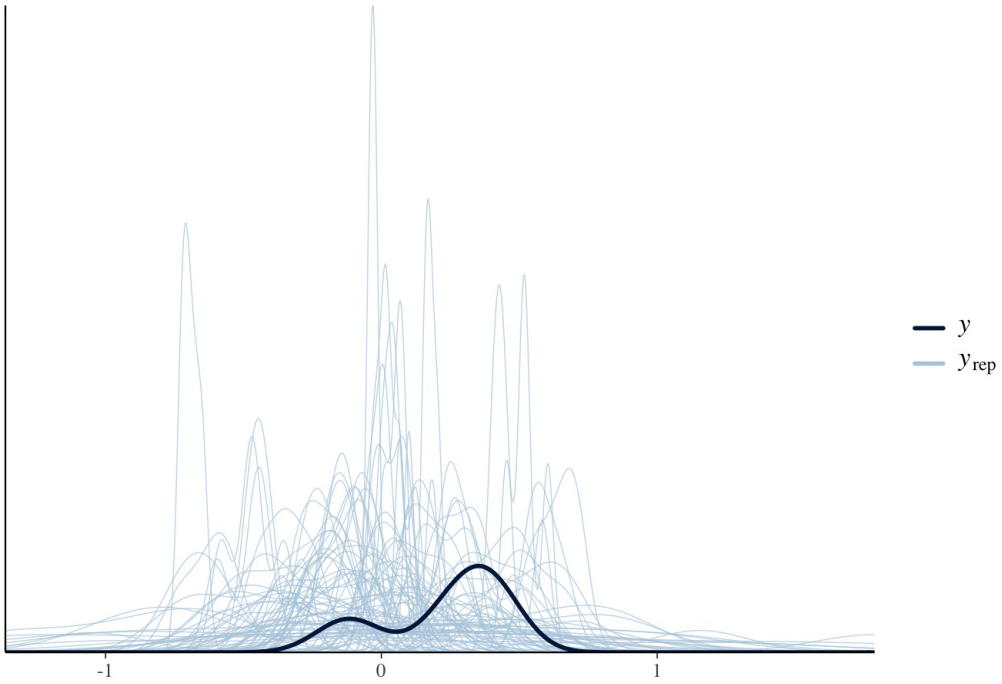
```
data_i <- Outcome_ES %>%  filter(!is.na(yi)) #keeping only the data that has yi values.
data_i <- data_i %>%  filter(!row_number() %in% c(2, 3))

model_p2_2_prior <- brm(
  pitch_data_f,
  data = data_i,
  family = gaussian,
  prior = model_2_priors,
  sample_prior = "only",
  backend = "cmdstanr",
  threads = threading(2),
  chains = 2,
  core = 2,
  control = list(adapt_delt = 0.99, max_treedepth = 20))
```

```
## Start sampling
```

```
## Running MCMC with 2 parallel 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 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 1 finished in 0.1 seconds.
## Chain 2 Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2 finished in 0.1 seconds.
##
## Both chains finished successfully.
## Mean chain execution time: 0.1 seconds.
## Total execution time: 0.2 seconds.
```

```
#update(model_p2_prior)
pp_check(model_p2_2_prior, ndraws=100) + labs(title = "Prior-predictive check, without influential studies")
```

Prior-predictive check, without influential studies



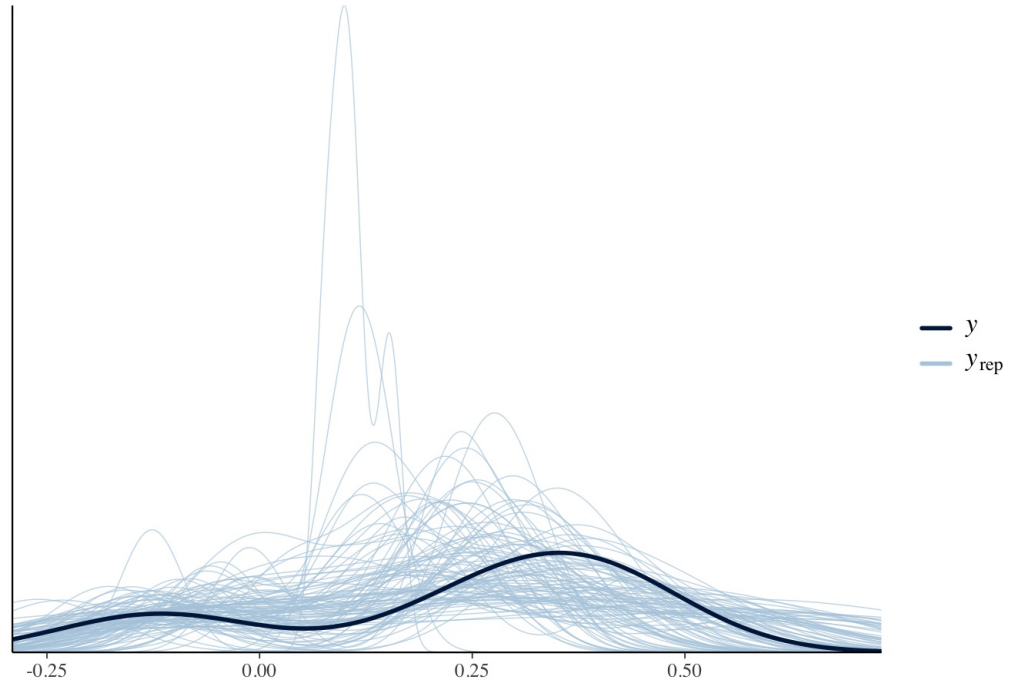

```
model_p2_2_post <- brm(
  pitch_data_f,
  data = data_i,
  family = gaussian,
  prior = model_2_priors,
  sample_prior = T,
  backend = "cmdstanr",
  threads = threading(2),
  chains = 2,
  core = 2,
  control = list(adapt_delt = 0.99, max_treedepth = 20))
```

```
## Start sampling
```

```
## Running MCMC with 2 parallel 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 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 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 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 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 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 1 Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1 Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 1 Iteration: 2000 / 2000 [100%]  (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 0.4 seconds.
## Chain 2 finished in 0.4 seconds.
##
## Both chains finished successfully.
## Mean chain execution time: 0.4 seconds.
## Total execution time: 0.5 seconds.
```

```
pp_check(model_p2_2_post, ndraws=100) + labs(title = "Posterior-predictive check, without influential studies")
```

Posterior-predictive check, without influential studies




```
#Outcome_ES
```