PESI analysis with brms

I. S. Plank

2024-10-02

Introduction

[!ADD info on study and task]

Some general settings

```r
# number of simulations
nsim = 500

# set number of iterations and warmup for models
iter = 4500
warm = 1500

# set the seed
set.seed(2468)
```

Package versions

```
## [1] "R version 4.4.1 (2024-06-14)"
```

```
## [1] "knitr version 1.46"
## [1] "ggplot2 version 3.5.1"
## [1] "brms version 2.21.0"
## [1] "designr version 0.1.13"
## [1] "bridgesampling version 1.1.2"
## [1] "tidyverse version 2.0.0"
## [1] "ggpubr version 0.6.0"
## [1] "ggrain version 0.0.4"
## [1] "bayesplot version 1.11.1"
## [1] "SBC version 0.3.0.9000"
## [1] "rstatix version 0.7.2"
## [1] "readODS version 2.3.0"
## [1] "BayesFactor version 0.9.12.4.7"
```

General info

We planned to determine the group-level effect subjects following Barr (2013). For each model, experiment specific priors were set based on previous literature or the task (see comments in the code).

We perform prior predictive checks as proposed in Schad, Betancourt and Vasishth (2020) using the SBC package. To do so, we create simulated datasets where parameters are simulated from the priors. These parameters are used to create one fake dataset. Both the true underlying parameters and the simulated values are saved.

Then, we create graphs showing the prior predictive distribution of the simulated discrimination threshold to check whether our priors fit our general expectations about the data. Next, we perform checks of computational faithfulness and model sensitivity as proposed by Schad, Betancourt and Vasishth (2020) and implemented in the SBC package. We create models for each of the simulated datasets. Last, we calculate performance metrics for each of these models, focusing on the population-level parameters.

Preparation and group comparisons

First, we load the data and combine it with demographic information including the diagnostic status of the subjects. Then, all predictors are set to sum contrasts. We have a look at the demographics describing our two diagnostic groups: autistic adults and adults without any neurological and psychiatric diagnoses.

Since this is sensitive data, we load the anonymised version of the processed data at this point but also leave the code we used to create it.

```r
# check if the data file exists, if yes load it:
if (!file.exists("PESI_data.RData")) {

  # set file paths
  fl.path = '/media/emba/emba-2/PESI'
  dt.path = paste(fl.path, 'BVET', sep = "/")

  # read in list of participants
  df.inc = read_ods(file.path(fl.path, "PESI_inc.ods"), range = "A1:K100") %>%
    select(subID, include_BV, include_ET)

  # create an anonymisation key
  df.inc = df.inc %>%
    mutate(
      PID   = subID,
      subID = as.numeric(as.factor(subID))
    )
  df.recode = df.inc %>% select(PID, subID) %>% distinct()
  recode = as.character(df.recode$subID)
  names(recode) = df.recode$PID

  # filter out pilots and cancelled testings
  df.inc = df.inc %>%
    filter(include_BV == 1)

  # load the relevant data in long format
  df.beh = list.files(path = dt.path, pattern = "PESI-BV", full.names = T) %>%
    map_df(~read_delim(., show_col_types = F, delim = ",",
                       col_types = "cddcccddddd")) %>%
    select(-dyad, -sync) %>%
    filter(subID %in% df.inc$PID) %>%
    group_by(subID) %>%
    mutate(
      no_trials = n()
    ) %>%
    # filter out participants where not the full task was recorded > no participants
    filter(no_trials == 64)

  # load demographic information to get diagnostic status
  df.sub = read_csv(file.path(dt.path, "PESI_centraXX.csv"), show_col_types = F) %>%
    mutate(
      diagnosis = recode(diagnosis, "CTR" = "COMP")
    )

  # load PST values
  df.pst = read_csv(file.path(dt.path, "PST_simult-params.csv"), show_col_types = F) %>%
    rename("subID" = "ID") %>% select(-group)

  # load the stimulus description file
  df.stm = read_csv(paste(fl.path, "PESI_videosel-full_230404.csv",
                          sep = "/"), show_col_types = F) %>%
    mutate(
      video = sprintf("PESI_%s_%08d", substr(dyad,1,3), frame_sta),
      dyad.type = case_match(context,
                             "homogeneous" ~ "non-autistic",
                             "heterogeneous" ~ "mixed")
    ) %>%
    select(video, dyad, sync, dyad.type, mot, peak)

  # merge together
  df = merge(df.beh, df.stm, all.x = T, by = "video") %>%
    mutate(
      # only use trials with confirmed rating and correct video duration
      use = if_else(confirmed == 1 & abs(dur-10) < 0.1, 1,0),
      rating.confirmed = if_else(use == 1, rating, NA)
           ) %>%
    arrange(subID, trl)

  # merge with group information and PST values
  df = merge(df.sub %>% select(subID, diagnosis), df, all.y = T) %>%
    merge(., df.pst, all.x = T) %>%
    mutate_if(is.character, as.factor)

  # check how many participants per group need to be excluded
  df.exc = df %>%
    group_by(subID, diagnosis) %>%
    summarise(
      total = sum(use)/64
    ) %>%
    filter(total <= 2/3) %>%
    group_by(diagnosis) %>%
    summarise(
      n = n()
    )

  # exclude participants with more than 33% of trials missing
  df = df %>%
    group_by(subID) %>%
    mutate(
      total = sum(use)/64
    ) %>%
    filter(total > 2/3)
  df.sub = df.sub %>%
    filter(subID %in% df$subID) %>%
    merge(., df.pst, all.x = T)

  # anonymise the data
  df$subID = str_replace_all(df$subID, recode)

  # load preprocessed eye tracking data and rename variables
  df.fix = readRDS(file.path(dt.path, "PESI-ET_agg.rds")) %>%
    rename(
      "trl" = "on_trialNo", "video" = "on_trialVid"
    )

  # anonymise ET data in the same way as the behavioural data
  df.fix$subID = str_replace_all(df.fix$subID, recode)

  # add information on the videos to the eye tracking data
  df.fix = merge(df.fix, df.stm, all.x = T) %>%
    # add diagnostic group
    merge(., df %>% select(subID, diagnosis) %>% distinct(), all.x = T) %>%
    mutate(across(where(is.character), as.factor))

  # get numbers of participants included in ET per group
  df.incET = df.fix %>%
    select(subID, diagnosis) %>%
    distinct() %>%
    group_by(diagnosis) %>%
    summarise(
      n = n()
    )

  # print gender frequencies and compare them across groups
  tb.gen = xtabs(~ gender + diagnosis, data = df.sub)
  ct.full = contingencyTableBF(tb.gen, sampleType = "indepMulti", fixedMargin = "cols")

  # check which outcomes of interest are normally distributed
  df.sht = df.sub %>%
    group_by(diagnosis) %>%
    shapiro_test(age, CFT_iq, BDI_total, STAITT_total, RAADS_total, ISH_total,
                 UI_total, sw_gz, sw_kl, thre, steep) %>%
    arrange(variable) %>%
    mutate(
      sig = if_else(p < 0.05, "*", "")
    )

  # some of the measures are not normally distributed;
  # therefore, we compute ranks for these outcomes
  df.sub = df.sub %>%
    mutate(
      rBDI   = rank(BDI_total),
      rISH   = rank(ISH_total),
      rRAADS = rank(RAADS_total),
      rUI    = rank(UI_total),
      rage   = rank(age),
```

```r
    rsw_kl = rank(sw_kl),
    rsteep = rank(steep),
    rthre  = rank(thre),
    diagnosis = as.factor(diagnosis)
  )

# now we can compute our ttests
ostt.age    = ttestBF(formula = rage         ~ diagnosis, data = df.sub)
ostt.iq     = ttestBF(formula = CFT_iq       ~ diagnosis, data = df.sub)
ostt.kl     = ttestBF(formula = sw_gz        ~ diagnosis, data = df.sub)
ostt.gz     = ttestBF(formula = rsw_kl       ~ diagnosis, data = df.sub)
ostt.STAITT = ttestBF(formula = STAITT_total ~ diagnosis, data = df.sub)
ostt.BDI    = ttestBF(formula = rBDI         ~ diagnosis, data = df.sub)
ostt.ISH    = ttestBF(formula = rISH         ~ diagnosis, data = df.sub)
ostt.RAADS  = ttestBF(formula = rRAADS       ~ diagnosis, data = df.sub)
ostt.UI     = ttestBF(formula = rUI          ~ diagnosis, data = df.sub)
ostt.thre   = ttestBF(formula = rthre        ~ diagnosis, data = df.sub)
ostt.steep  = ttestBF(formula = rsteep       ~ diagnosis, data = df.sub)

# ...and put everything in a new dataframe for printing
measurement  = "Age"
ASD      = sprintf("%.2f (±%.2f)",
                   mean(df.sub[df.sub$diagnosis == "ASD",]$age),
                   sd(df.sub[df.sub$diagnosis == "ASD",]$age)/
                     sqrt(nrow(df.sub[df.sub$diagnosis == "ASD",])))
COMP      = sprintf("%.2f (±%.2f)",
                    mean(df.sub[df.sub$diagnosis == "COMP",]$age),
                    sd(df.sub[df.sub$diagnosis == "COMP",]$age)/
                      sqrt(nrow(df.sub[df.sub$diagnosis == "COMP",])))
logBF10 = sprintf("%.3f", ostt.age@bayesFactor[["bf"]])
df.table = data.frame(measurement, ASD, COMP, logBF10)
df.table = rbind(df.table,
                 c(
                   "BDI",
                   sprintf("%.2f (±%.2f)",
                           mean(df.sub[df.sub$diagnosis == "ASD",]$BDI_total),
                           sd(df.sub[df.sub$diagnosis == "ASD",]$BDI_total)/
                             sqrt(nrow(df.sub[df.sub$diagnosis == "ASD",]))),
                   sprintf("%.2f (±%.2f)",
                           mean(df.sub[df.sub$diagnosis == "COMP",]$BDI_total),
                           sd(df.sub[df.sub$diagnosis == "COMP",]$BDI_total)/
                             sqrt(nrow(df.sub[df.sub$diagnosis == "COMP",]))),
                   sprintf("%.3f", ostt.BDI@bayesFactor[["bf"]])
                 ),
                 c(
                   "Gender (diverse/agender/non-binary - female - male)",
                   sprintf("%d - %d - %d",
                           nrow(df.sub[df.sub$diagnosis == "ASD" &
                                         df.sub$gender == "dan",]),
                           nrow(df.sub[df.sub$diagnosis == "ASD" &
                                         df.sub$gender == "fem",]),
                           nrow(df.sub[df.sub$diagnosis == "ASD" &
                                         df.sub$gender == "mal",])),
                   sprintf("%d - %d - %d",
                           nrow(df.sub[df.sub$diagnosis == "COMP" &
                                         df.sub$gender == "dan",]),
                           nrow(df.sub[df.sub$diagnosis == "COMP" &
                                         df.sub$gender == "fem",]),
                           nrow(df.sub[df.sub$diagnosis == "COMP" &
                                         df.sub$gender == "mal",])),
                   sprintf("%.3f", ct.full@bayesFactor[["bf"]])
                 ),
                 c(
                   "IQ",
                   sprintf("%.2f (±%.2f)",
                           mean(df.sub[df.sub$diagnosis == "ASD",]$CFT_iq),
                           sd(df.sub[df.sub$diagnosis == "ASD",]$CFT_iq)/
                             sqrt(nrow(df.sub[df.sub$diagnosis == "ASD",]))),
                   sprintf("%.2f (±%.2f)",
                           mean(df.sub[df.sub$diagnosis == "COMP",]$CFT_iq),
                           sd(df.sub[df.sub$diagnosis == "COMP",]$CFT_iq)/
                             sqrt(nrow(df.sub[df.sub$diagnosis == "COMP",]))),
                   sprintf("%.3f", ostt.iq@bayesFactor[["bf"]])
                 ),
                 c(
                   "RAADS",
                   sprintf("%.2f (±%.2f)",
                           mean(df.sub[df.sub$diagnosis == "ASD",]$RAADS_total),
                           sd(df.sub[df.sub$diagnosis == "ASD",]$RAADS_total)/
                             sqrt(nrow(df.sub[df.sub$diagnosis == "ASD",]))),
                   sprintf("%.2f (±%.2f)",
                           mean(df.sub[df.sub$diagnosis == "COMP",]$RAADS_total),
                           sd(df.sub[df.sub$diagnosis == "COMP",]$RAADS_total)/
                             sqrt(nrow(df.sub[df.sub$diagnosis == "COMP",]))),
                   sprintf("%.3f", ostt.RAADS@bayesFactor[["bf"]])
                 ),
                 c(
                   "D2 - concentration performance",
                   sprintf("%.2f (±%.2f)",
                           mean(df.sub[df.sub$diagnosis == "ASD",]$sw_kl),
                           sd(df.sub[df.sub$diagnosis == "ASD",]$sw_kl)/
                             sqrt(nrow(df.sub[df.sub$diagnosis == "ASD",]))),
                   sprintf("%.2f (±%.2f)",
                           mean(df.sub[df.sub$diagnosis == "COMP",]$sw_kl),
                           sd(df.sub[df.sub$diagnosis == "COMP",]$sw_kl)/
                             sqrt(nrow(df.sub[df.sub$diagnosis == "COMP",]))),
                   sprintf("%.3f", ostt.kl@bayesFactor[["bf"]])
                 ),
                 c(
                   "D2 - speed",
                   sprintf("%.2f (±%.2f)",
                           mean(df.sub[df.sub$diagnosis == "ASD",]$sw_gz),
                           sd(df.sub[df.sub$diagnosis == "ASD",]$sw_gz)/
                             sqrt(nrow(df.sub[df.sub$diagnosis == "ASD",]))),
                   sprintf("%.2f (±%.2f)",
                           mean(df.sub[df.sub$diagnosis == "COMP",]$sw_gz),
                       sd(df.sub[df.sub$diagnosis == "COMP",]$sw_gz)/
                         sqrt(nrow(df.sub[df.sub$diagnosis == "COMP",]))),
               sprintf("%.3f", ostt.gz@bayesFactor[["bf"]])
             ),
             c(
               "STAI-trait",
               sprintf("%.2f (±%.2f)",
                       mean(df.sub[df.sub$diagnosis == "ASD",]$STAITT_total),
                       sd(df.sub[df.sub$diagnosis == "ASD",]$STAITT_total)/
                         sqrt(nrow(df.sub[df.sub$diagnosis == "ASD",]))),
               sprintf("%.2f (±%.2f)",
                       mean(df.sub[df.sub$diagnosis == "COMP",]$STAITT_total),
                       sd(df.sub[df.sub$diagnosis == "COMP",]$STAITT_total)/
                         sqrt(nrow(df.sub[df.sub$diagnosis == "COMP",]))),
               sprintf("%.3f", ostt.STAITT@bayesFactor[["bf"]])
             ),
             c(
               "Ishihara",
               sprintf("%.2f (±%.2f)",
                       mean(df.sub[df.sub$diagnosis == "ASD",]$ISH_total),
                       sd(df.sub[df.sub$diagnosis == "ASD",]$ISH_total)/
                         sqrt(nrow(df.sub[df.sub$diagnosis == "ASD",]))),
               sprintf("%.2f (±%.2f)",
                       mean(df.sub[df.sub$diagnosis == "COMP",]$ISH_total),
                       sd(df.sub[df.sub$diagnosis == "COMP",]$ISH_total)/
                         sqrt(nrow(df.sub[df.sub$diagnosis == "COMP",]))),
               sprintf("%.3f", ostt.ISH@bayesFactor[["bf"]])
             ),
             c(
               "UI",
               sprintf("%.2f (±%.2f)",
                       mean(df.sub[df.sub$diagnosis == "ASD",]$UI_total),
                       sd(df.sub[df.sub$diagnosis == "ASD",]$UI_total)/
                         sqrt(nrow(df.sub[df.sub$diagnosis == "ASD",]))),
               sprintf("%.2f (±%.2f)",
                       mean(df.sub[df.sub$diagnosis == "COMP",]$UI_total),
                       sd(df.sub[df.sub$diagnosis == "COMP",]$UI_total)/
                         sqrt(nrow(df.sub[df.sub$diagnosis == "COMP",]))),
               sprintf("%.3f", ostt.UI@bayesFactor[["bf"]])
             ),
             c(
               "PST - threshold",
               sprintf("%.2f (±%.2f)",
                       mean(df.sub[df.sub$diagnosis == "ASD",]$thre),
                       sd(df.sub[df.sub$diagnosis == "ASD",]$thre)/
                         sqrt(nrow(df.sub[df.sub$diagnosis == "ASD",]))),
               sprintf("%.2f (±%.2f)",
                       mean(df.sub[df.sub$diagnosis == "COMP",]$thre),
                       sd(df.sub[df.sub$diagnosis == "COMP",]$thre)/
                         sqrt(nrow(df.sub[df.sub$diagnosis == "COMP",]))),
               sprintf("%.3f", ostt.thre@bayesFactor[["bf"]])
             ),
             c(
                      "PST - steepness",
                      sprintf("%.2f (±%.2f)",
                              mean(df.sub[df.sub$diagnosis == "ASD",]$steep),
                              sd(df.sub[df.sub$diagnosis == "ASD",]$steep)/
                                sqrt(nrow(df.sub[df.sub$diagnosis == "ASD",]))),
                      sprintf("%.2f (±%.2f)",
                              mean(df.sub[df.sub$diagnosis == "COMP",]$steep),
                              sd(df.sub[df.sub$diagnosis == "COMP",]$steep)/
                                sqrt(nrow(df.sub[df.sub$diagnosis == "COMP",]))),
                      sprintf("%.3f", ostt.steep@bayesFactor[["bf"]])
                    )
                 ) %>% arrange(measurement)

  # save it all
  df = df %>% select(subID, diagnosis, dyad, video, run, trl, sync, rating,
                      dyad.type, mot, peak, rating.confirmed, thre, steep)
  save(df, df.table, df.sht, ct.full, df.exc, df.incET, df.fix,
       file = "PESI_data.RData")

} else {

  load("PESI_data.RData")

}

# print the group of included participants
kable(df %>% select(subID, diagnosis) %>% distinct() %>% group_by(diagnosis) %>% count())
```

diagnosis	n
ASD	27
COMP	36

```
# print the group of excluded participants
kable(df.exc)
```

diagnosis	n
ASD	5
COMP	2

```
rm(df.exc)

# print number of included participants in eye tracking
kable(df.incET)
```

diagnosis	n
ASD	17
COMP	25
NA	3

```r
rm(df.incET)

# print the outcome of the shapiro tests
kable(df.sht)
```

diagnosis	variable	statistic	p	sig
ASD	BDI_total	0.8733786	0.0034673	*
COMP	BDI_total	0.8548415	0.0002465	*
ASD	CFT_iq	0.9306520	0.0717023	
COMP	CFT_iq	0.9464526	0.0808711	
ASD	ISH_total	0.4654390	0.0000000	*
COMP	ISH_total	0.4929020	0.0000000	*
ASD	RAADS_total	0.8323572	0.0005256	*
COMP	RAADS_total	0.8278027	0.0000611	*
ASD	STAITT_total	0.9502740	0.2176972	
COMP	STAITT_total	0.9638854	0.2824299	
ASD	UI_total	0.9090163	0.0216386	*
COMP	UI_total	0.9019167	0.0038513	*
ASD	age	0.9308549	0.0725292	
COMP	age	0.9395161	0.0490695	*
ASD	steep	0.4008972	0.0000000	*
COMP	steep	0.5898618	0.0000000	*
ASD	sw_gz	0.9456727	0.1680787	
COMP	sw_gz	0.9540447	0.1401784	
ASD	sw_kl	0.9567579	0.3110060	
COMP	sw_kl	0.9353969	0.0365896	*
ASD	thre	0.8004993	0.0001796	*
COMP	thre	0.9466602	0.1261984	

```r
rm(df.sht)

# print the outcome of the contingency table
ct.full@bayesFactor
```

```
##                                 bf error                     time         code
## Non-indep. (a=1) -0.7457339     0 Sat Sep 28 14:21:48 2024 24c64947f5b5
```

```r
# aggregate the data due to large differences between videos
df.agg = df %>%
  group_by(subID, diagnosis, dyad, sync, dyad.type) %>%
  summarise(
    rating.confirmed = mean(rating.confirmed, na.rm = T)
  ) %>% ungroup() %>%
  mutate_if(is.character, as.factor)

# set and print the contrasts
contrasts(df.agg$sync) = contr.sum(2)
contrasts(df.agg$sync)
```

```
##      [,1]
## high    1
## low    -1
```

```r
contrasts(df.agg$dyad.type) = contr.sum(2)
contrasts(df.agg$dyad.type)
```

```
##               [,1]
## mixed            1
## non-autistic    -1
```

```r
contrasts(df.agg$diagnosis) = contr.sum(2)
contrasts(df.agg$diagnosis)
```

```
##      [,1]
## ASD     1
## COMP   -1
```

```r
contrasts(df.fix$sync) = contr.sum(2)
contrasts(df.fix$sync)
```

```
##      [,1]
## high    1
## low    -1
```

```r
contrasts(df.fix$dyad.type) = contr.sum(2)
contrasts(df.fix$dyad.type)
```

```
##               [,1]
## mixed            1
## non-autistic    -1
```

```r
contrasts(df.fix$diagnosis) = contr.sum(2)
contrasts(df.fix$diagnosis)
```

```
##      [,1]
## ASD     1
## COMP   -1
```

```r
# print final group comparisons for the paper
kable(df.table)
```

measurement	ASD	COMP	logBF10
Age	28.59 (±1.34)	27.69 (±0.94)	-1.303
BDI	17.44 (±2.48)	4.86 (±0.82)	12.160
D2 - concentration performance	101.74 (±2.65)	104.28 (±1.67)	-0.982
D2 - speed	102.04 (±2.45)	104.69 (±1.64)	-0.939
Gender (diverse/agender/non-binary - female - male)	0 - 11 - 16	0 - 19 - 17	-0.746
IQ	113.59 (±4.05)	114.58 (±2.16)	-1.327
Ishihara	28.96 (±0.46)	28.25 (±0.57)	-0.190
PST - steepness	NA (±NA)	NA (±NA)	-1.348
PST - threshold	NA (±NA)	NA (±NA)	-1.240
RAADS	31.19 (±1.61)	7.44 (±0.93)	28.408
STAI-trait	52.15 (±2.22)	30.89 (±1.60)	18.780
UI	65.63 (±2.55)	37.97 (±1.75)	21.418

The two diagnostic groups are similar in age, concentration and speed (D2), colorblindness score (Ishihara), IQ and gender distribution as well as on a perceptual simultaneity task (PST). However, they seem to differ in their questionnaire scores measuring depressive symptoms (BDI), autism (RAADS), trait anxiety (STAI-trait) and intolerance of uncertainty (UI).

Ratings

To achieve good posterior fit, we aggregated the rating data. First, we attempted to run a model including all main effects as slopes as well as the interaction of sync and dyad type for subjects. However, all models including random slopes had major divergence issues. Even the model only including only the intercepts on the group-level had some divergent transitions. Nonetheless, we opted to include both the intercept for the dyads and the subjects, and look out for problems with the actual model.

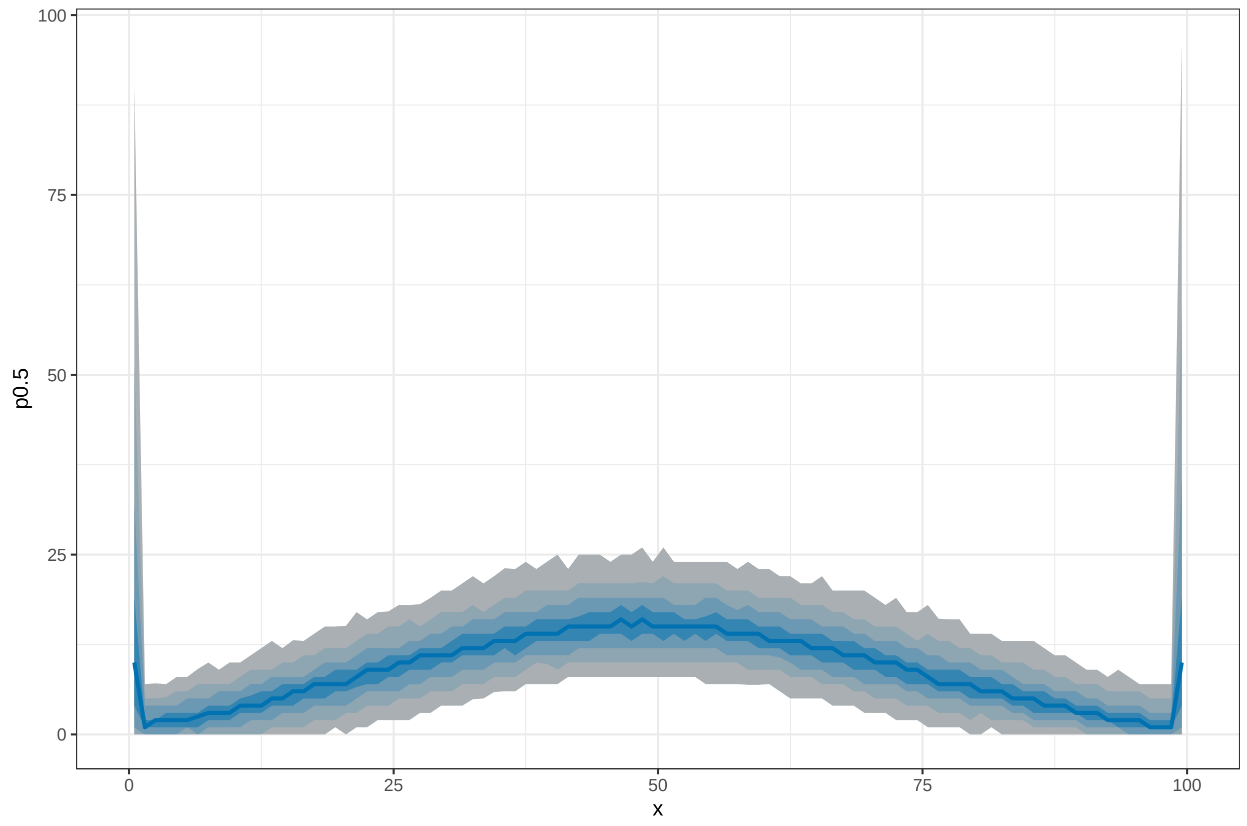
Model of aggregated ratings

```r
# set formula considering all combinations
code   = "PESI_int"
f.pesi = brms::bf(rating.confirmed ~ diagnosis * sync * dyad.type
                  + (1 | subID) + (1 | dyad))

# set weakly informed priors
priors = c(
  prior(normal(50,  10), class = Intercept),
  prior(normal(0,   10), class = sigma),
  prior(normal(0,   10), class = sd),
  #prior(lkj(2),          class = cor),
  # differences due to dyad.type
  prior(normal(-5,   5), class = b, coef = dyad.type1), # mixed
  # differences due to synchrony
  prior(normal(5,    5), class = b, coef = sync1), # high
  # effect of synchrony decreased when mixed dyad
  prior(normal(-5,   5), class = b, coef = sync1:dyad.type1),
  # effect of dyad type decreased in autistic subjects
  prior(normal(-5,   5), class = b, coef = diagnosis1:dyad.type1),
  # no specific expectations for diagnostic groups and other interactions
  prior(normal(0,    5), class = b)
)

if (file.exists(file.path(cache_dir, paste0("df_res_", code, ".rds")))) {
  # load in the results of the SBC
  df.results = readRDS(file.path(cache_dir, paste0("df_res_", code, ".rds")))
  df.backend = readRDS(file.path(cache_dir, paste0("df_div_", code, ".rds")))
  dat        = readRDS(file.path(cache_dir, paste0("dat_", code, ".rds")))
} else {
  # set the seed
  set.seed(2469)
  # create the data
  gen  = SBC_generator_brms(f.pesi, data = df.agg, prior = priors,
                            thin = 50, warmup = 20000, refresh = 2000
  )
  dat  = generate_datasets(gen, nsim)
  saveRDS(dat, file = sprintf("%s/dat_%s.rds", cache_dir, code))

  # perform the SBC
  bck = SBC_backend_brms_from_generator(gen, chains = 4, thin = 1,
                                        warmup = warm, iter = iter,
                                        inits = 0.1)
  res = compute_SBC(dat, bck,
                    cache_mode = "results",
                    cache_location = file.path(cache_dir, sprintf("res_%s", code)))
  # save the results dataframes
  df.results = res$stats
  df.backend = res$backend_diagnostics
  saveRDS(df.results, file = file.path(cache_dir, paste0("df_res_", code, ".rds")))
  saveRDS(df.backend, file = file.path(cache_dir, paste0("df_div_", code, ".rds")))

}
```

We start by investigating the rhats and the number of divergent samples. This shows that 14 of 500 simulations had at least one parameter that had an rhat of at least 1.05, and 186 models had divergent samples (mean number of samples of the simulations with divergent samples: 7.85). We will have to watch out for this.

Next, we can plot the simulated values to perform prior predictive checks.

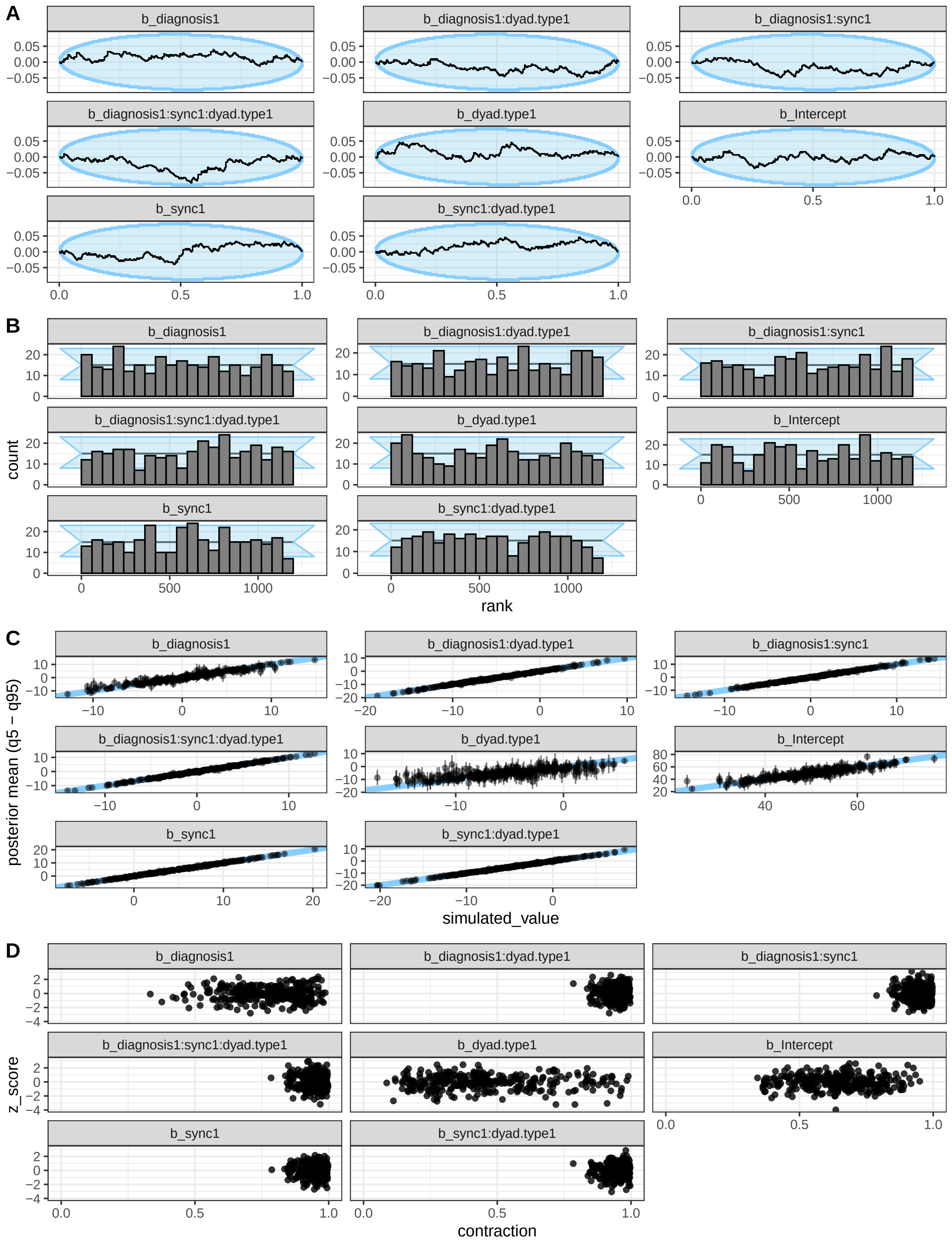
```
# create a matrix out of generated data
dvname = gsub(" ", "", gsub("[\\|~].*", "", f.pesi)[1])
dvfakemat = matrix(NA, nrow(dat[['generated']][[1]]), length(dat[['generated']]))
for (i in 1:length(dat[['generated']])) {
  dvfakemat[,i] = dat[['generated']][[i]][[dvname]]
}
truePars = dat$variables

# plot simulated data for prior predictive checks
dvmax = 100
dvfakematH = dvfakemat;
dvfakematH[dvfakematH > dvmax] = dvmax
dvfakematH[dvfakematH < 0] = 0
breaks = seq(0, max(dvfakematH, na.rm=T), length.out = 100)
binwidth = round(breaks[2] - breaks[1])
breaks = seq(0, max(dvfakematH), by = binwidth)
histmat = matrix(NA, ncol = dim(dvfakematH)[2] + binwidth, nrow = length(breaks)-1)
for (i in 1:dim(dvfakematH)[2]) {
  histmat[,i] = hist(dvfakematH[,i], breaks = breaks, plot = F)$counts
}
probs = seq(0.1, 0.9, 0.1)
quantmat= as.data.frame(matrix(NA, nrow=dim(histmat)[1], ncol = length(probs)))
names(quantmat) = paste0("p", probs)
for (i in 1:dim(histmat)[1]) {
  quantmat[i,] = quantile(histmat[i,], p = probs, na.rm = T)
}
quantmat$x = breaks[2:length(breaks)] - binwidth/2 # add bin mean
ggplot(data = quantmat, aes(x = x)) +
  geom_ribbon(aes(ymax = p0.9, ymin = p0.1), fill = c_light) +
  geom_ribbon(aes(ymax = p0.8, ymin = p0.2), fill = c_light_highlight) +
  geom_ribbon(aes(ymax = p0.7, ymin = p0.3), fill = c_mid) +
  geom_ribbon(aes(ymax = p0.6, ymin = p0.4), fill = c_mid_highlight) +
  geom_line(aes(y = p0.5), colour = c_dark, linewidth = 1) +
  xlim(0, max(dvfakematH)) +
  theme_bw()
```



[!ADD]

```
# get simulation numbers with issues
rank = max(df.results$max_rank)
check = merge(df.results %>%
    group_by(sim_id) %>%
      summarise(
        rhat = max(rhat, na.rm = T),
        mean_rank = mean(max_rank)
        ) %>%
    filter(rhat >= 1.05 | mean_rank != rank),
  df.backend %>% filter(n_divergent > 0), all = T)

# plot SBC with functions from the SBC package focusing on population-level parameters
df.results.b = df.results %>%
  filter(substr(variable, 1, 2) == "b_") %>%
  filter(!(sim_id %in% check$sim_id))
p1 = plot_ecdf_diff(df.results.b) + theme_bw() + theme(legend.position = "none") +
  scale_x_continuous(breaks=scales::pretty_breaks(n = 3)) +
  scale_y_continuous(breaks=scales::pretty_breaks(n = 3))
p2 = plot_rank_hist(df.results.b, bins = 20) + theme_bw() +
  scale_x_continuous(breaks=scales::pretty_breaks(n = 3)) +
  scale_y_continuous(breaks=scales::pretty_breaks(n = 3))
p3 = plot_sim_estimated(df.results.b, alpha = 0.5) + theme_bw() +
  scale_x_continuous(breaks=scales::pretty_breaks(n = 3)) +
  scale_y_continuous(breaks=scales::pretty_breaks(n = 3))
p4 = plot_contraction(
  df.results.b,
  prior_sd = setNames(c(10,
                       rep(5, length(unique(df.results.b$variable))-1)),
                     unique(df.results.b$variable))) +
  theme_bw() +
  scale_x_continuous(breaks=scales::pretty_breaks(n = 3)) +
  scale_y_continuous(breaks=scales::pretty_breaks(n = 3))

p = ggarrange(p1, p2, p3, p4, labels = "AUTO", ncol = 1, nrow = 4)
annotate_figure(p,
                top = text_grob("Computational faithfulness and model sensitivity",
                face = "bold", size = 14))
```

Computational faithfulness and model sensitivity



Second, we check the ranks of the parameters. If the model is unbiased, these should be uniformly distributed

(Schad, Betancourt and Vasishth, 2020). The sample empirical cumulative distribution function (ECDF) lies within the theoretical distribution (95%) and the rank histogram also shows ranks within the 95% expected range, although there are some small deviations. We judge this to be acceptable.

Third, we investigated the relationship between the simulated true parameters and the posterior estimates. Although there are individual values diverging from the expected pattern, most parameters were recovered successfully within an uncertainty interval of alpha = 0.05.

Last, we explore the z-score and the posterior contraction of our population-level predictors. The z-score "determines the distance of the posterior mean from the true simulating parameter", while the posterior contraction "estimates how much prior uncertainty is reduced in the posterior estimation" (Schad, Betancourt and Vasisth, 2020). Both look acceptable.

Posterior predictive checks

As the next step, we fit the model and check whether the chains have converged, which they seem to have. We then perform posterior predictive checks on the model using the bayesplot package.

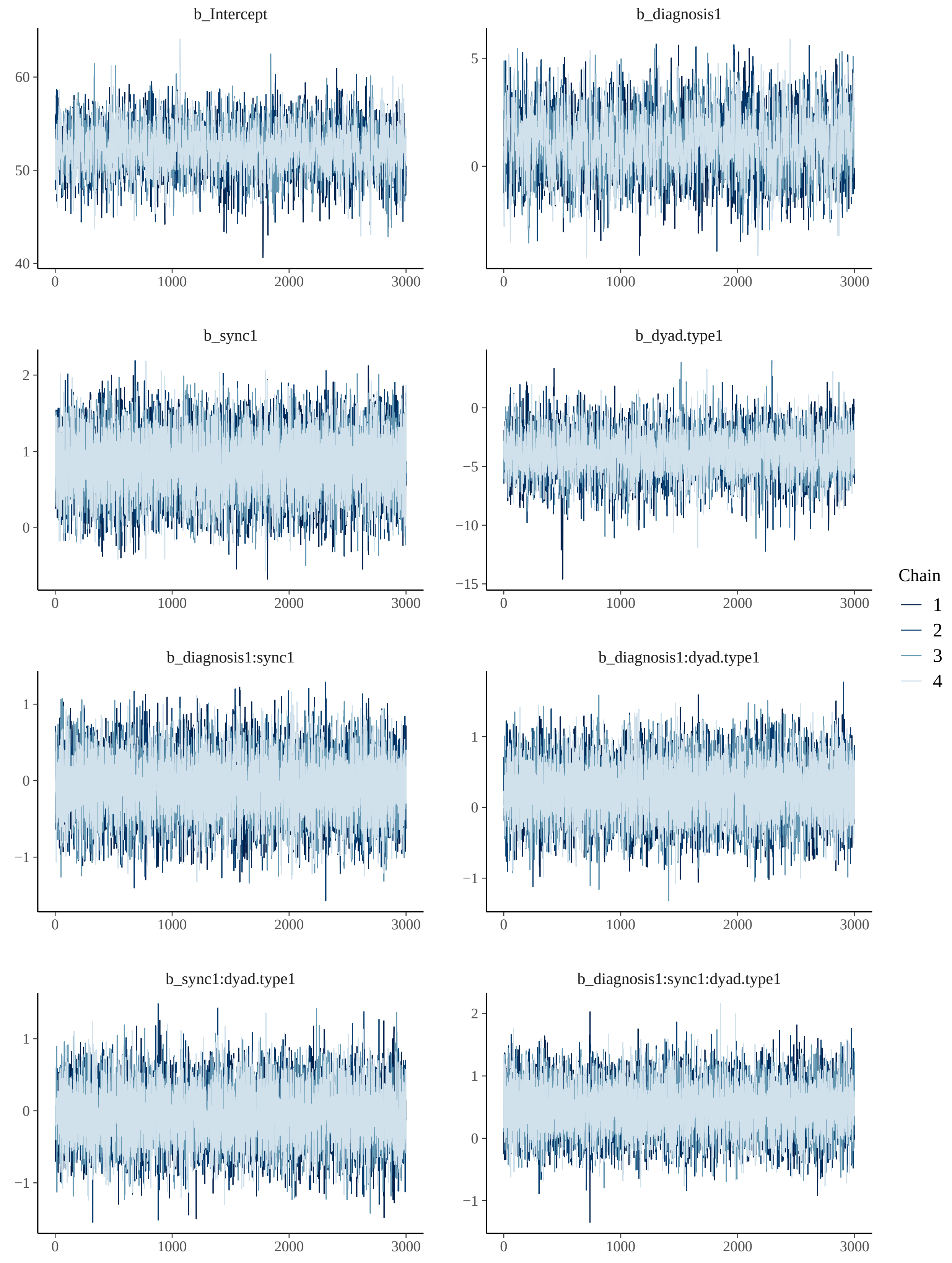
```r
# fit the maximal model
set.seed(2486)
m.pesi = brm(f.pesi,
            df.agg, prior = priors,
            iter = iter, warmup = warm,
            backend = "cmdstanr", threads = threading(8),
            file = "m_PESI",
            save_pars = save_pars(all = TRUE)
            )

# in this model, there are no divergent samples
sum(subset(nuts_params(m.pesi), Parameter == "divergent__")$Value)
```

```
## [1] 0
```

```r
# check that rhats are below 1.01
sum(brms::rhat(m.pesi) >= 1.01, na.rm = T)
```

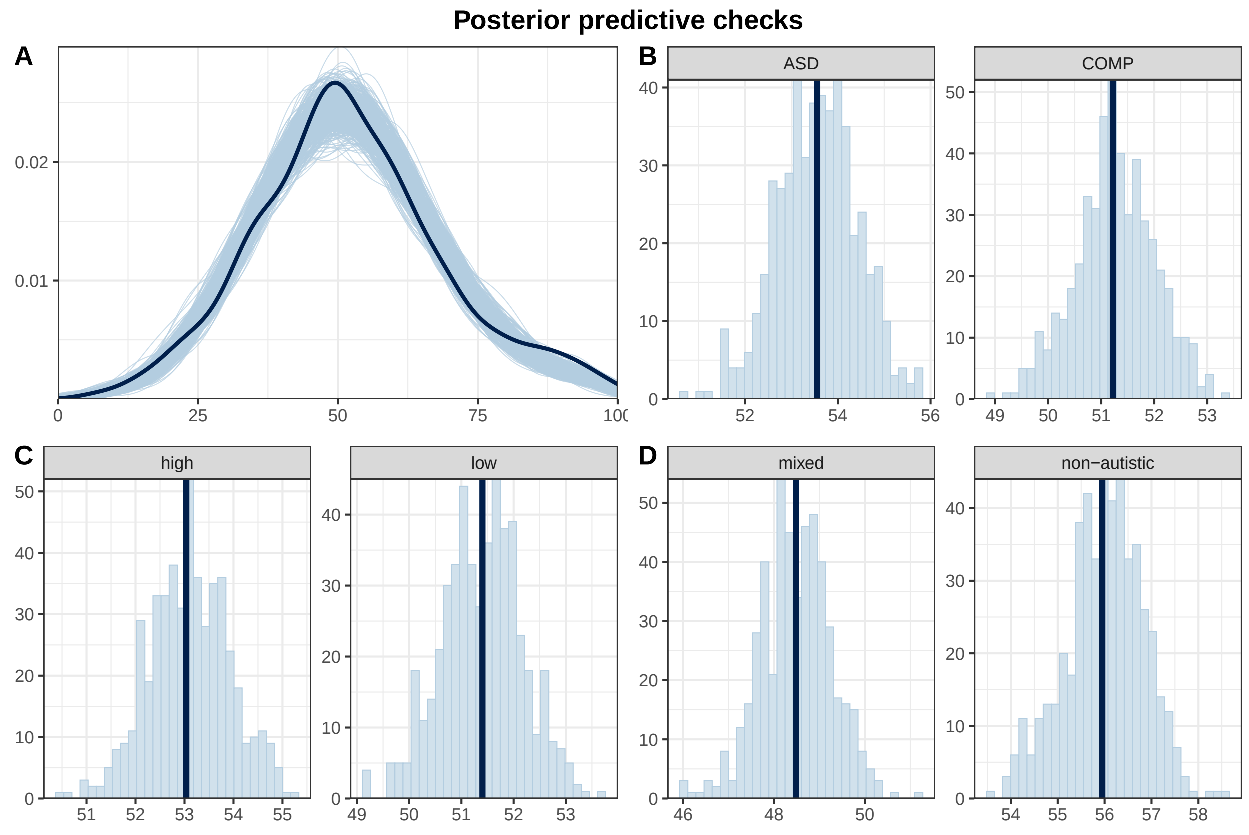
```
## [1] 0
```

```r
# and the chains have converged
post.draws = as_draws_df(m.pesi)
mcmc_trace(post.draws, regex_pars = "^b_",
           facet_args = list(ncol = 2)) +
  scale_x_continuous(breaks=scales::pretty_breaks(n = 3)) +
  scale_y_continuous(breaks=scales::pretty_breaks(n = 3))
```



This model has no divergent samples, and no rhat that is higher or equal to 1.01. Therefore, we go ahead and perform our posterior predictive checks.

```
# get the posterior predictions
post.pred = posterior_predict(m.pesi, ndraws = nsim)

# check the fit of the predicted data compared to the real data
p1 = pp_check(m.pesi, ndraws = nsim) +
  theme_bw() + theme(legend.position = "none") + xlim(0,100)

# distributions of means and sds compared to the real values per group
p2 = ppc_stat_grouped(df.agg$rating.confirmed, post.pred, df.agg$diagnosis) +
  theme_bw() + theme(legend.position = "none")
# ... sync level
p3 = ppc_stat_grouped(df.agg$rating.confirmed, post.pred, df.agg$sync) +
  theme_bw() + theme(legend.position = "none")
# ... and dyad type
p4 = ppc_stat_grouped(df.agg$rating.confirmed, post.pred, df.agg$dyad.type) +
  theme_bw() + theme(legend.position = "none")

p = ggarrange(p1, p2, p3, p4,
          nrow = 2, ncol = 2, labels = "AUTO")
annotate_figure(p,
                top = text_grob("Posterior predictive checks",
                face = "bold", size = 14))
```



The predictions based on the model capture the data very well. The means for each group are firmly distributed around the real values. This further increased our trust in the model and we move on to interpret it.

Model summary

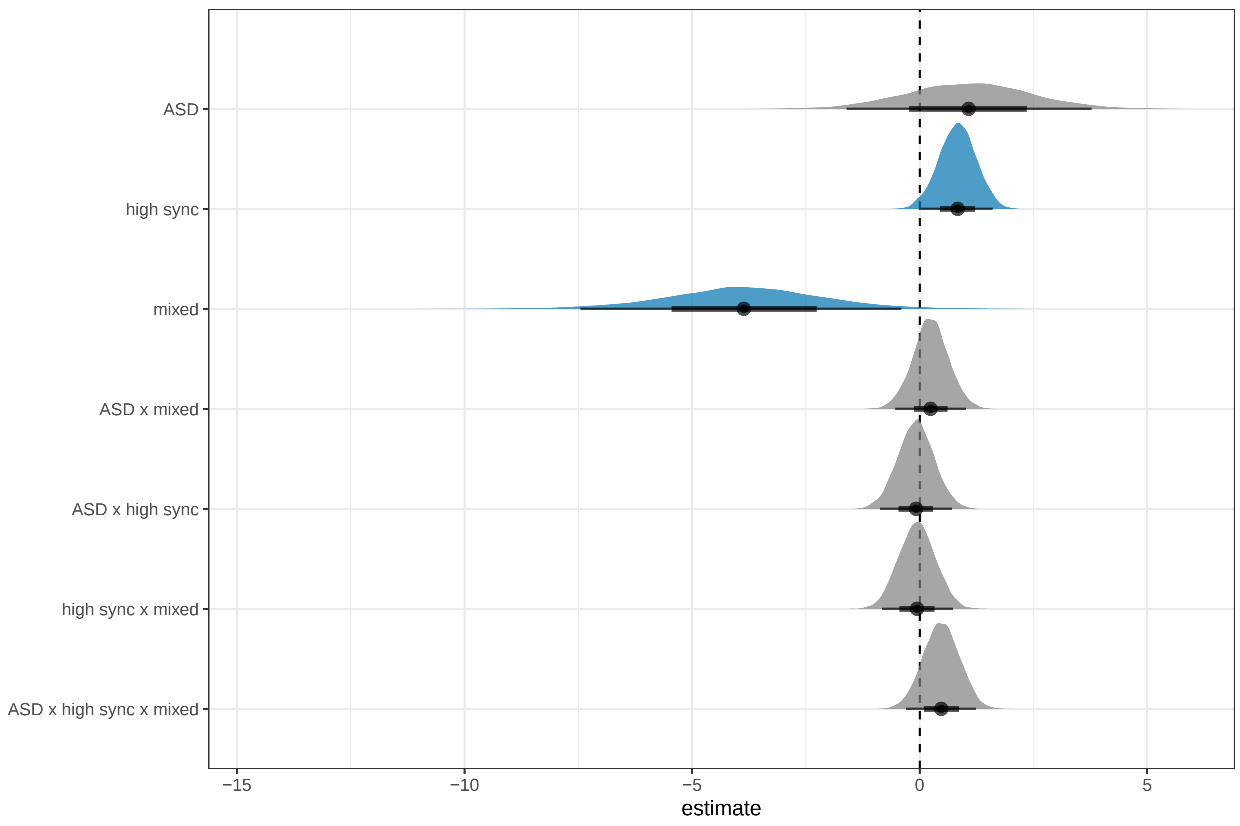
Now that we are convinced that we can trust our model, we have a look at the model and its estimates.

```
# print a summary
summary(m.pesi)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: rating.confirmed ~ diagnosis * sync * dyad.type + (1 | subID) + (1 | dyad)
##    Data: df.agg (Number of observations: 1008)
##   Draws: 4 chains, each with iter = 4500; warmup = 1500; thin = 1;
##          total post-warmup draws = 12000
##
## Multilevel Hyperparameters:
## ~dyad (Number of levels: 8)
##               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)     5.18      1.78     2.77     9.62 1.00     4251     6415
##
## ~subID (Number of levels: 63)
##               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)    10.74      1.07     8.87    13.00 1.00     3107     5369
##
## Regression Coefficients:
##                              Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## Intercept                       52.24      2.32    47.65    56.84 1.00     2930
## diagnosis1                       1.07      1.37    -1.61     3.78 1.00     2137
## sync1                            0.83      0.40     0.02     1.60 1.00    22391
## dyad.type1                      -3.87      1.78    -7.45    -0.40 1.00     5143
## diagnosis1:sync1                -0.08      0.40    -0.87     0.71 1.00    22927
## diagnosis1:dyad.type1            0.24      0.39    -0.54     1.02 1.00    21273
## sync1:dyad.type1                -0.06      0.40    -0.83     0.73 1.00    22302
## diagnosis1:sync1:dyad.type1      0.48      0.40    -0.30     1.24 1.00    21095
##                              Tail_ESS
## Intercept                        5158
## diagnosis1                       4105
## sync1                            7815
## dyad.type1                       6811
## diagnosis1:sync1                 8372
## diagnosis1:dyad.type1            8337
## sync1:dyad.type1                 8364
## diagnosis1:sync1:dyad.type1      8549
##
## Further Distributional Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma    12.56      0.29    12.00    13.15 1.00    17960     9039
##
## 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).
```

```
# plot the posterior distributions
post.draws %>%
```

```r
  select(starts_with("b_")) %>%
  pivot_longer(cols = starts_with("b_"),
               names_to = "coef",
               values_to = "estimate") %>%
  subset(!startsWith(coef, "b_Int")) %>%
  mutate(
    coef = substr(coef, 3, nchar(coef)),
    coef = str_replace_all(coef, ":", " x "),
    coef = str_replace_all(coef, "diagnosis1", "ASD"),
    coef = str_replace_all(coef, "sync1", "high sync"),
    coef = str_replace_all(coef, "dyad.type1", "mixed"),
    coef_order = case_when(
      coef == "ASD" ~ 100,
      coef == "high sync" ~ 99,
      coef == "mixed" ~ 98,
      T ~ 100-nchar(coef)),
    coef = fct_reorder(coef, coef_order)
  ) %>%
  group_by(coef) %>%
  mutate(
    cred = case_when(
      (mean(estimate) < 0 & quantile(estimate, probs = 0.975) < 0) |
        (mean(estimate) > 0 & quantile(estimate, probs = 0.025) > 0) ~ "credible",
      T ~ "not credible"
    )
  ) %>% ungroup() %>%
  ggplot(aes(x = estimate, y = coef, fill = cred)) +
  geom_vline(xintercept = 0, linetype = 'dashed') +
  ggdist::stat_halfeye(alpha = 0.7) + ylab(NULL) + theme_bw() +
  scale_fill_manual(values = c(credible = c_dark, c_light)) +
  theme(legend.position = "none")
```



Inferences

```
# H1.1 Context: Social interactions of no-diagnosis non-autistic dyads are
# rated more positively than mixed-diagnosis dyads consisting of one autistic
# and one non-autistic interaction partner.
h1.1 = hypothesis(m.pesi, "dyad.type1 < 0")
h1.1
```

```
## Hypothesis Tests for class b:
##          Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio Post.Prob
## 1 (dyad.type1) < 0    -3.87      1.78    -6.79    -1.01      59.91      0.98
##   Star
## 1    *
## ---
## 'CI': 90%-CI for one-sided and 95%-CI for two-sided hypotheses.
## '*': For one-sided hypotheses, the posterior probability exceeds 95%;
## for two-sided hypotheses, the value tested against lies outside the 95%-CI.
## Posterior probabilities of point hypotheses assume equal prior probabilities.
```

```
# H1.2 Synchrony: Social interactions with high interpersonal synchrony of
# motion energy are rated more positively than social interactions with low
# interpersonal synchrony of motion energy.
h1.2 = hypothesis(m.pesi, "sync1 > 0")
h1.2
```

```
## Hypothesis Tests for class b:
##    Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio Post.Prob Star
## 1 (sync1) > 0     0.83       0.4     0.16     1.49      43.78      0.98    *
## ---
## 'CI': 90%-CI for one-sided and 95%-CI for two-sided hypotheses.
## '*': For one-sided hypotheses, the posterior probability exceeds 95%;
## for two-sided hypotheses, the value tested against lies outside the 95%-CI.
## Posterior probabilities of point hypotheses assume equal prior probabilities.
```

```
# H1.3 Diagnostic status: Ratings of social interactions differ between
# autistic and non-autistic participants.
h1.3 = hypothesis(m.pesi, "diagnosis1 > 0")
h1.3
```

```
## Hypothesis Tests for class b:
##         Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio Post.Prob
## 1 (diagnosis1) > 0     1.07      1.37     -1.2     3.34       3.67      0.79
##   Star
## 1
## ---
## 'CI': 90%-CI for one-sided and 95%-CI for two-sided hypotheses.
## '*': For one-sided hypotheses, the posterior probability exceeds 95%;
## for two-sided hypotheses, the value tested against lies outside the 95%-CI.
## Posterior probabilities of point hypotheses assume equal prior probabilities.
```

```
# H1.4 Synchrony x dyad type: The effect of interpersonal motion synchrony
# on ratings is decreased for mixed-diagnosis dyads compared to no-diagnosis dyads.
h1.4 = hypothesis(m.pesi, "sync1:dyad.type1 < 0")
h1.4
```

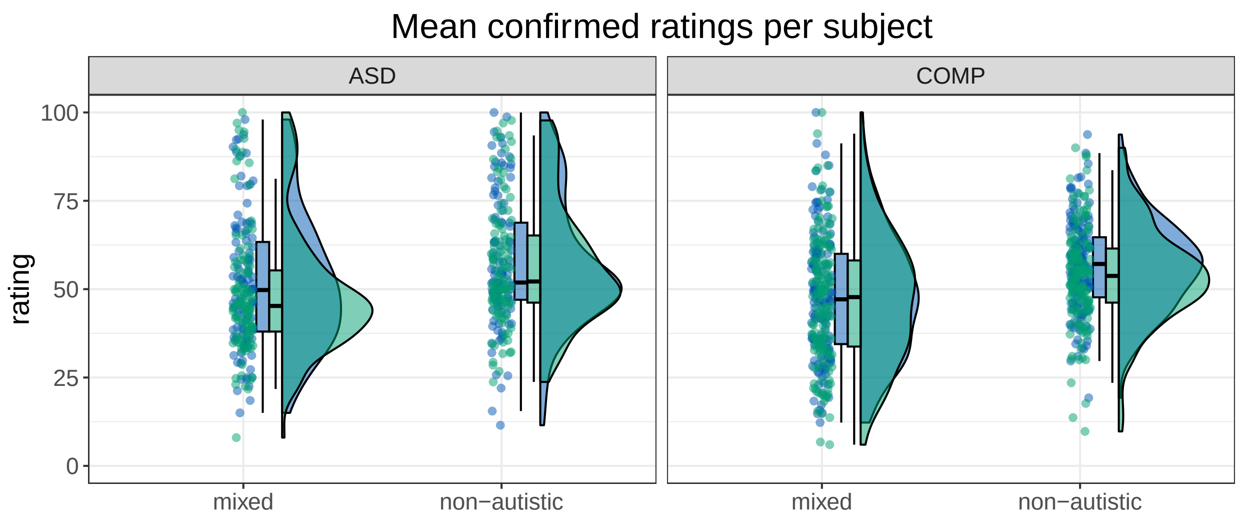
```
## Hypothesis Tests for class b:
##                Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio
## 1 (sync1:dyad.type1) < 0    -0.06       0.4    -0.71      0.6       1.25
##   Post.Prob Star
## 1      0.56
## ---
## 'CI': 90%-CI for one-sided and 95%-CI for two-sided hypotheses.
## '*': For one-sided hypotheses, the posterior probability exceeds 95%;
## for two-sided hypotheses, the value tested against lies outside the 95%-CI.
## Posterior probabilities of point hypotheses assume equal prior probabilities.
```

```
# H1.5 Dyad type x diagnostic status: The effect of dyad type on ratings is
# decreased in autistic compared to non-autistic participants.
h1.5 = hypothesis(m.pesi, "diagnosis1:dyad.type1 < 0")
h1.5
```

```
## Hypothesis Tests for class b:
##                  Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio
## 1 (diagnosis1:dyad.... < 0     0.24      0.39    -0.41     0.89       0.35
##   Post.Prob Star
## 1      0.26
## ---
## 'CI': 90%-CI for one-sided and 95%-CI for two-sided hypotheses.
## '*': For one-sided hypotheses, the posterior probability exceeds 95%;
## for two-sided hypotheses, the value tested against lies outside the 95%-CI.
## Posterior probabilities of point hypotheses assume equal prior probabilities.
```

Plots

```r
# rain cloud plot for ratings
df.agg %>%
  mutate(
    dyad = as.factor(dyad)
  ) %>%
  ggplot(aes(dyad.type, rating.confirmed, fill = sync,
             colour = sync, fill = sync)) +
  geom_rain(rain.side = 'r',
            boxplot.args = list(colour = "black",
                                outlier.shape = NA,
                                show_guide = FALSE,
                                alpha = 0.5),
            violin.args  = list(colour = "black",
                                outlier.shape = NA,
                                show_guide = FALSE,
                                alpha = .5),
            point.args   = list(show_guide = FALSE,
                                alpha = .5),
            boxplot.args.pos = list(
              position =
                ggpp::position_dodgenudge(x = .1, width = 0.1),
              width = 0.1
            )) +
  ylim(0, 100) +
  facet_wrap(. ~ diagnosis) +
  scale_fill_manual(values = custom.col) +
  scale_color_manual(values = custom.col) +
  labs(title = "Mean confirmed ratings per subject",
       x = "",
       y = "rating") +
  theme_bw() +
  theme(legend.position = "bottom",
        plot.title = element_text(hjust = 0.5),
        legend.direction = "horizontal",
        text = element_text(size = 15))
```



Fixation durations