S1: behavioural analysis with brms

I. S. Plank

2025-07-02

# S1.1 Introduction

This R Markdown script analyses behavioural data from the FAB (face attention bias) paradigm of the EMBA project. The data was preprocessed before being read into this script.

The task is modeled after Jakobsen et al. (2021), *Attention, Perception, & Psychophysics* and the authors were kind enough to share their stimuli. Each trial starts with a black fixation cross on a white background. Then, a cue consisting of a pair of pictures, one object and one face, is shown with one picture on the left and one on the right of the previous location of the fixation cross. In line with Moore et al. (2012), *J Autism Dev Disord*, we set the duration of the cue presentation to 200ms. Afterwards, a target square appears either at the previous location of the face or the object. Subjects task is to determine the location (right or left) of the target as fast and accurate as possible. The target only disappears when the participant gives their answer.

The visual angle of the target was 1.17 degrees, the visual angle of the cues was 4.25 and the distance of the centre of the target and cue from the fixation cross was 2.67 degrees.

## Some general settings

# number of simulations  
nsim = 250  
  
# set number of iterations and warmup for models  
iter = 3000  
warm = 1000  
  
# set the seed  
set.seed(2468)

## Package versions

The following packages are used in this RMarkdown file:

## [1] "R version 4.5.0 (2025-04-11)"

## [1] "knitr version 1.50"  
## [1] "ggplot2 version 3.5.2"  
## [1] "brms version 2.22.0"  
## [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] "vtable version 1.4.8"  
## [1] "ggrain version 0.0.4"  
## [1] "bayesplot version 1.12.0"  
## [1] "SBC version 0.3.0.9000"  
## [1] "rstatix version 0.7.2"  
## [1] "flextable version 0.9.8"  
## [1] "officer version 0.6.10"  
## [1] "BayesFactor version 0.9.12.4.7"  
## [1] "effectsize version 1.0.1"  
## [1] "bayestestR version 0.16.0"

## 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 performed prior predictive checks as proposed in Schad, Betancourt and Vasishth (2020) using the SBC package based on the original design with three groups. To do so, we create 250 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 discrimination 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. We did not rerun SBC after adding the exploratory sample of ADHD+ASD.

We base our assessment of the hypothesis on the posterior distributions. Therefore, we perform posterior prdictive checks and in some cases simplify the model by aggregating values to improve posterior fit.

## 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 four diagnostic groups: adults with ADHD, autistic adults, autistic adults with ADHD (explorative) 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.

# check if the data file exists, if yes load it:  
if (!file.exists("FAB\_data.RData")) {  
  
 # get demo info for subjects  
 df.sub = read\_csv(file.path("/home/emba/Documents/EMBA/CentraXX", "EMBA\_centraXX.csv"),   
 show\_col\_types = F) %>%  
 mutate(  
 diagnosis = recode(diagnosis, "CTR" = "COMP"),  
 adhd.meds.desc = adhd.meds,  
 adhd.meds = if\_else(is.na(adhd.meds), FALSE, TRUE)  
 )  
   
 # set the data path  
 dt.path = "/home/emba/Documents/EMBA/BVET"  
 dt.explo = "/home/emba/Documents/EMBA/BVET-explo"  
   
 # load excluded participants (low accuracy)  
 exc = c(scan(file.path(dt.path, 'FAB\_exc.txt'), what="character", sep=NULL),  
 scan(file.path(dt.explo, 'FAB\_exc.txt'), what="character", sep=NULL))  
 df.exc = df.sub %>% filter(subID %in% exc) %>%   
 select(diagnosis) %>%   
 group\_by(diagnosis) %>% count()  
   
 # load the behavioral data and merge with group  
 df.fab = merge(df.sub %>% select(subID, diagnosis, RAADS\_total, ASRS\_total, adhd.meds),   
 readRDS(file = paste0(dt.path, '/df\_FAB.RDS')), all.y = T) %>%  
 mutate\_if(is.character, as.factor)  
 df.exp = merge(df.sub %>% select(subID, diagnosis, RAADS\_total, ASRS\_total, adhd.meds),   
 readRDS(file = paste0(dt.explo, '/df\_FAB.RDS')), all.y = T) %>%  
 mutate\_if(is.character, as.factor)  
   
 # only keep participants included in the study in the subject data frame  
 subIDs = as.character(c(unique(df.fab$subID), unique(df.exp$subID)))  
 df.sub = df.sub %>% filter(subID %in% subIDs)  
   
 df.med = df.sub %>% group\_by(diagnosis) %>%  
 summarise(  
 adhd.meds = mean(adhd.meds)  
 )  
   
 adhd.meds.desc = unique(df.sub[!is.na(df.sub$adhd.meds.desc),]$adhd.meds.desc)  
   
 # load the eye tracking data and only keep participants included in the study,  
 # so no people with more than 33% mistakes, no people without any saccades   
 # and no people with too many blinks  
 df.sac = rbind(readRDS(file.path(dt.explo, "FAB\_ET\_data.rds")),  
 readRDS(file.path(dt.path, "FAB\_ET\_data.rds"))) %>%  
 merge(., df.sub %>% select(subID, diagnosis), keep.y = T)  
   
 # check groups of people who had no relevant saccades at all  
 df.nosac = df.sac %>% filter(is.na(trl)) %>%  
 group\_by(diagnosis) %>%  
 count()  
  
 # anonymise the data  
 df.fab = df.fab %>%  
 mutate(  
 PID = subID,  
 subID = as.factor(as.numeric(subID))  
 )  
 df.exp = df.exp %>%  
 mutate(  
 PID = subID,  
 subID = as.factor(as.numeric(subID) + length(unique(df.fab$subID)))  
 )  
   
 # get a correspondence of original PIDs and anonymised subIDs  
 df.recode = rbind(df.fab %>% select(PID, subID) %>% distinct(),  
 df.exp %>% select(PID, subID) %>% distinct())  
 recode = as.character(df.recode$subID)  
 names(recode) = df.recode$PID  
 df.fab = df.fab %>% select(-PID)  
 df.exp = df.exp %>% select(-PID)  
   
 # anonymise ET data in the same way  
 df.sac$subID = str\_replace\_all(df.sac$subID, recode)  
   
 # 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")  
 # since only DAN in the ADHD group, we try again after excluding them  
 ct.mf = contingencyTableBF(tb.gen[2:3,],   
 sampleType = "indepMulti",   
 fixedMargin = "cols")  
 # we add this information to our demographics tablerbind(df.demo,   
 df.demo = data.frame(  
 measurement = "Gender",  
 ADHD = sprintf("%.0f - %.0f - %.0f",   
 tb.gen["fem","ADHD"],   
 tb.gen["mal","ADHD"],  
 tb.gen["dan","ADHD"]  
 ),  
 `ADHD+ASD` = sprintf("%.0f - %.0f - %.0f",   
 tb.gen["fem","BOTH"],   
 tb.gen["mal","BOTH"],  
 tb.gen["dan","BOTH"]  
 ),  
 ASD = sprintf("%.0f - %.0f - %.0f",   
 tb.gen["fem","ASD"],   
 tb.gen["mal","ASD"],  
 tb.gen["dan","ASD"]  
 ),  
 COMP = sprintf("%.0f - %.0f - %.0f",   
 tb.gen["fem","COMP"],   
 tb.gen["mal","COMP"],  
 tb.gen["dan","COMP"]  
 ),  
 bf.log = ct.full@bayesFactor[["bf"]]  
 )  
   
 # then, we save some more gender information in a table in case we need it  
 tb.gen = xtabs(~ gender + diagnosis + cis, data = df.sub)  
   
 # get the gender descriptions of the not-male and not-female participants  
 gen.desc = unique(tolower(df.sub[df.sub$gender == "dan",]$gender\_desc))  
   
 # convert the measures to long which we include in the participant table  
 df.sublng = df.sub %>%  
 # rename some of the variables  
 rename(  
 "RADS-R" = "RAADS\_total",  
 "ASRS-v1.1" = "ASRS\_total",  
 "Age" = "age",  
 "IQ estimate" = "iq",  
 "Education" = "edu"  
 ) %>%  
 select(diagnosis, Age, `IQ estimate`, `ASRS-v1.1`, `RADS-R`, Education) %>%  
 pivot\_longer(cols = where(is.numeric)) %>%  
 mutate\_if(is.character, as.factor)  
   
 # initialise the data frame for posthoc tests  
 df.post = data.frame()  
   
 # now we loop through our measurements to create our demographics table  
 for (m in unique(df.sublng$name)) {  
 # select the relevant part of df.sub  
 df.rel = df.sublng %>% filter(name == m)  
 # check which of the group's data is not normally distributed  
 df.sht = df.rel %>%   
 group\_by(diagnosis) %>%  
 shapiro\_test(value) %>%  
 filter(p < 0.05)  
 # if more than zero is not normally distributed...  
 if (nrow(df.sht) > 0) {  
 # rank transform the data  
 df.rel = df.rel %>% ungroup() %>% mutate(value = rank(value))  
 }  
 # compute the ANOVA  
 aov = anovaBF(value ~ diagnosis, data = df.rel)  
 # get back the original, untransformed values   
 df.rel = df.sublng %>% filter(name == m)  
 # put all the information into the demographics table  
 df.demo = rbind(df.demo,   
 data.frame(  
 measurement = m,  
 ADHD = sprintf("%.2f ±%.2f (%.0f to %.0f)",   
 # ignore NAs because edu missing for one person  
 mean(df.rel[df.rel$diagnosis == "ADHD",]$value, na.rm = T),   
 sd(df.rel[df.rel$diagnosis == "ADHD",]$value, na.rm = T)/  
 sum(df.rel$diagnosis == "ADHD"),   
 min(df.rel[df.rel$diagnosis == "ADHD",]$value, na.rm = T),   
 max(df.rel[df.rel$diagnosis == "ADHD",]$value, na.rm = T)  
 ),  
 `ADHD+ASD` = sprintf("%.2f ±%.2f (%.0f to %.0f)",   
 mean(df.rel[df.rel$diagnosis == "BOTH",]$value),   
 sd(df.rel[df.rel$diagnosis == "BOTH",]$value)/  
 sum(df.rel$diagnosis == "BOTH"),   
 min(df.rel[df.rel$diagnosis == "BOTH",]$value),   
 max(df.rel[df.rel$diagnosis == "BOTH",]$value)  
 ),  
 ASD = sprintf("%.2f ±%.2f (%.0f to %.0f)",   
 mean(df.rel[df.rel$diagnosis == "ASD",]$value),   
 sd(df.rel[df.rel$diagnosis == "ASD",]$value)/  
 sum(df.rel$diagnosis == "ASD"),   
 min(df.rel[df.rel$diagnosis == "ASD",]$value),   
 max(df.rel[df.rel$diagnosis == "ASD",]$value)  
 ),  
 COMP = sprintf("%.2f ±%.2f (%.0f to %.0f)",   
 mean(df.rel[df.rel$diagnosis == "COMP",]$value),   
 sd(df.rel[df.rel$diagnosis == "COMP",]$value)/  
 sum(df.rel$diagnosis == "COMP"),   
 min(df.rel[df.rel$diagnosis == "COMP",]$value),   
 max(df.rel[df.rel$diagnosis == "COMP",]$value)  
 ),  
 bf.log = aov@bayesFactor[["bf"]]  
 ))  
   
 # next, we want to check whether there are group differences  
 if (abs(exp(aov@bayesFactor$bf)) > 3) {  
 # do the group comparisons   
 aov.ADHDvASD = anovaBF(value ~ diagnosis,   
 data = df.rel %>% filter(diagnosis %in% c("ADHD", "ASD")))  
 aov.ADHDvBOTH = anovaBF(value ~ diagnosis,   
 data = df.rel %>% filter(diagnosis %in% c("ADHD", "BOTH")))  
 aov.ADHDvCOMP = anovaBF(value ~ diagnosis,   
 data = df.rel %>% filter(diagnosis %in% c("ADHD", "COMP")))  
 aov.ASDvBOTH = anovaBF(value ~ diagnosis,   
 data = df.rel %>% filter(diagnosis %in% c("ASD", "BOTH")))  
 aov.ASDvCOMP = anovaBF(value ~ diagnosis,   
 data = df.rel %>% filter(diagnosis %in% c("ASD", "COMP")))  
 aov.BOTHvCOMP = anovaBF(value ~ diagnosis,   
 data = df.rel %>% filter(diagnosis %in% c("BOTH", "COMP")))  
 # put into the posthoc data frame  
 df.post = rbind(df.post,   
 data.frame(  
 measurement = m,  
 ADHDvASD = aov.ADHDvASD@bayesFactor[["bf"]],  
 ADHDvBOTH = aov.ADHDvBOTH@bayesFactor[["bf"]],  
 ADHDvCOMP = aov.ADHDvCOMP@bayesFactor[["bf"]],  
 ASDvBOTH = aov.ASDvBOTH@bayesFactor[["bf"]],  
 ASDvCOMP = aov.ASDvCOMP@bayesFactor[["bf"]],  
 BOTHvCOMP = aov.BOTHvCOMP@bayesFactor[["bf"]]  
 ))  
 }  
 }  
   
 # save the demographics and the posthoc table as word documents  
 read\_docx() %>%  
 body\_add\_table(df.demo %>% arrange(measurement) %>%   
 mutate(bf.log =   
 if\_else(  
 bf.log > 3,   
 sprintf("%.3f\*", bf.log),  
 sprintf("%.3f", bf.log)))) %>%  
 print(target = "FAB\_demo.docx")  
 read\_docx() %>%  
 body\_add\_table(df.post %>%   
 mutate\_if(is.numeric,   
 ~ifelse(.>3,sprintf("%.3f\*", .),sprintf("%.3f", .)))) %>%  
 print(target = "FAB\_post.docx")  
   
 # check how many of each group are above threshold for asrs and rads  
 tb.screen = xtabs(~ diagnosis + screening,   
 data = df.sub %>%  
 select(diagnosis, ASRS\_screen, RAADS\_total) %>%  
 mutate(  
 screening = case\_when(  
 ASRS\_screen >= 4 & RAADS\_total > 81 ~ "screenBOTH",  
 ASRS\_screen >= 4 & RAADS\_total <= 81 ~ "screenADHD",  
 ASRS\_screen < 4 & RAADS\_total <= 81 ~ "screenNone",  
 ASRS\_screen < 4 & RAADS\_total > 81 ~ "screenASD"  
 )  
 ))  
   
 # save it all  
 save(df.fab, df.sac, df.exp, ct.full, ct.mf, df.exc, tb.screen,  
 df.nosac, gen.desc, tb.gen, adhd.meds.desc, df.med,  
 file = "FAB\_data.RData")  
   
} else {  
   
 load("FAB\_data.RData")  
   
}  
  
# print the group of excluded participants based on low accuracy (< 2/3)  
kable(df.exc)

| diagnosis | n |
| --- | --- |
| ADHD | 1 |
| ASD | 1 |

rm(df.exc)  
  
# print the group of the participants included in behavioural and eye tracking   
kable(merge(  
 df.sac %>% filter(!is.na(trl)) %>% select(subID, diagnosis) %>% distinct() %>%   
 group\_by(diagnosis) %>% summarise(`sample size eye tracking` = n()),  
 rbind(df.fab, df.exp) %>% select(subID, diagnosis) %>% distinct() %>%   
 group\_by(diagnosis) %>% summarise(`sample size behavioural` = n())  
 ))

| diagnosis | sample size eye tracking | sample size behavioural |
| --- | --- | --- |
| ADHD | 16 | 23 |
| ASD | 19 | 24 |
| BOTH | 22 | 23 |
| COMP | 21 | 24 |

# Note: eye-tracking only collected if calibration accuracy < 0.5, then exclusion:  
# 1 due to more than 1/3 blinks   
# 2 due to no relevant saccades  
# how many have been removed due to no relevant saccades?  
kable(df.nosac)

| diagnosis | n |
| --- | --- |
| ASD | 1 |
| COMP | 1 |

rm(df.nosac)  
  
# print the how many people are above threshold for clinical self assessment  
kable(tb.screen)

|  | screenADHD | screenASD | screenBOTH | screenNone |
| --- | --- | --- | --- | --- |
| ADHD | 4 | 2 | 11 | 6 |
| ASD | 1 | 17 | 5 | 1 |
| BOTH | 0 | 8 | 15 | 0 |
| COMP | 1 | 3 | 0 | 20 |

# print the outcome of the two contingency tables for comparison  
# based on full sample with agender, diverse and non-binary in one category  
ct.full@bayesFactor

## bf error time code  
## Non-indep. (a=1) -4.405795 0 Wed Mar 12 12:18:46 2025 3b78a3d905301

# based on female and male participants only  
ct.mf@bayesFactor

## bf error time code  
## Non-indep. (a=1) -2.737336 0 Wed Mar 12 12:18:46 2025 3b78a7b1eb2ca

# combine explorative and original data  
df.fab = rbind(df.fab, df.exp)  
  
# set the levels of the diagnosis factor  
df.fab$diagnosis = factor(df.fab$diagnosis,   
 levels = c("ADHD", "ASD", "BOTH", "COMP"))  
  
# set and print the contrasts  
contrasts(df.fab$cue) = contr.sum(2)  
contrasts(df.fab$cue)

## [,1]  
## face 1  
## object -1

contrasts(df.fab$diagnosis) = contr.sum(4)  
contrasts(df.fab$diagnosis)

## [,1] [,2] [,3]  
## ADHD 1 0 0  
## ASD 0 1 0  
## BOTH 0 0 1  
## COMP -1 -1 -1

The three diagnostic groups are similar in age, IQ and gender distribution. However, they seem to differ in their questionnaire scores measuring ADHD (ASRS), depression (BDI), autism (RAADS) and alexithymia (TAS).

# S1.2 Reaction times

First, we analyse the reaction times for all correctly answered trials to assess whether participants answer faster if the target appears at the previous location of the face, which we refer to as face attention bias (FAB). In our preregistration, we formulated the following hypotheses:

H1a) COMP participants react faster in response to targets appearing on the side of the face compared to targets appearing on the side of the object (face attention bias; Jakobsen et al., 2021). H1b) ADHD participants react slower than COMP participants in both cue conditions (Sonuga-Barke et al., 2004). H1c) ASD participants react slower than COMP participants in both cue conditions (Ghosn et al., 2018). H1d) Face attention bias is decreased in ASD participants compared to COMP participants (Moore et al., 2012). H1e) Face attention bias in ADHD participants differs from face attention bias in COMP participants.

## Full model

### Simulation-based calibration

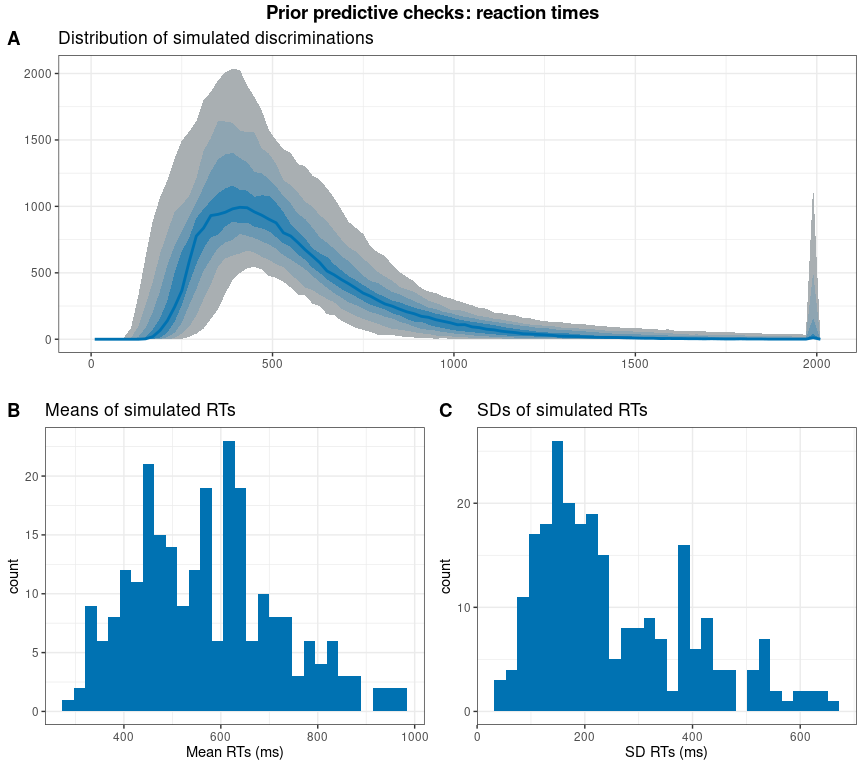
First, we attempted to use a full model for the data. This model includes multiple instances of each stimulus per participant in each of the conditions (face cue or object cue). Therefore, we need slopes for the cue per subject as well as for cue, diagnosis and their interaction for the stimulus.

code = "FAB"  
  
# full model formula  
f.fab = brms::bf(rt.cor ~ diagnosis \* cue + (cue | subID) + (cue \* diagnosis | stm) )  
  
# set informed priors based on previous results  
priors = c(  
 # general priors based on SBV  
 prior(normal(6, 0.3), class = Intercept),  
 prior(normal(0, 0.5), class = sigma),  
 prior(normal(0, 0.1), class = sd),  
 prior(lkj(2), class = cor),  
 # face attention bias effect based on Jakobsen et al. (2021)  
 prior(normal(-0.01, 0.04), class = b, coef = cue1),  
 # ADHD subjects being slower based on Pievsky & McGrath (2018)  
 prior(normal(0.025, 0.04), class = b, coef = diagnosis1),  
 # ASD subjects being slower based on Morrison et al. (2018)  
 prior(normal(0.025, 0.04), class = b, coef = diagnosis2),  
 # decreased FAB in ASD subjects based on Moore et al. (2012)  
 prior(normal(0.01, 0.04), class = b, coef = diagnosis2:cue1),  
 # no specific expectations for FAB in ADHD  
 prior(normal(0, 0.04), class = b),  
 # shift  
 prior(normal(200, 100), class = ndt)  
)  
  
# check if the SBC already exists  
if (file.exists(file.path(cache\_dir, sprintf("df\_res\_%s.rds", code)))) {  
 # load in the results of the SBC  
 df.results = readRDS(file.path(cache\_dir, sprintf("df\_res\_%s.rds", code)))  
 df.backend = readRDS(file.path(cache\_dir, sprintf("df\_div\_%s.rds", code)))  
 dat = readRDS(file.path(cache\_dir, sprintf("dat\_%s.rds", code)))  
} else {  
 # perform the SBC  
 gen = SBC\_generator\_brms(f.fab, data = df.fab, prior = priors,  
 family = shifted\_lognormal,  
 thin = 50, warmup = 20000, refresh = 2000)  
 bck = SBC\_backend\_brms\_from\_generator(gen, chains = 4, thin = 1,  
 init = 0.1, warmup = warm, iter = iter)  
 dat = generate\_datasets(gen, nsim)  
 saveRDS(dat, file.path(cache\_dir, sprintf("dat\_%s.rds", code)))  
 res = compute\_SBC(dat,   
 bck,  
 cache\_mode = "results",   
 cache\_location = file.path(cache\_dir, sprintf("res\_%s", code)))  
 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 6 of 250 simulations had at least one parameter that had an rhat of at least 1.05, and 1 model had divergent samples (mean number of samples of the simulations with divergent samples: 4). This suggests that this model performs well.

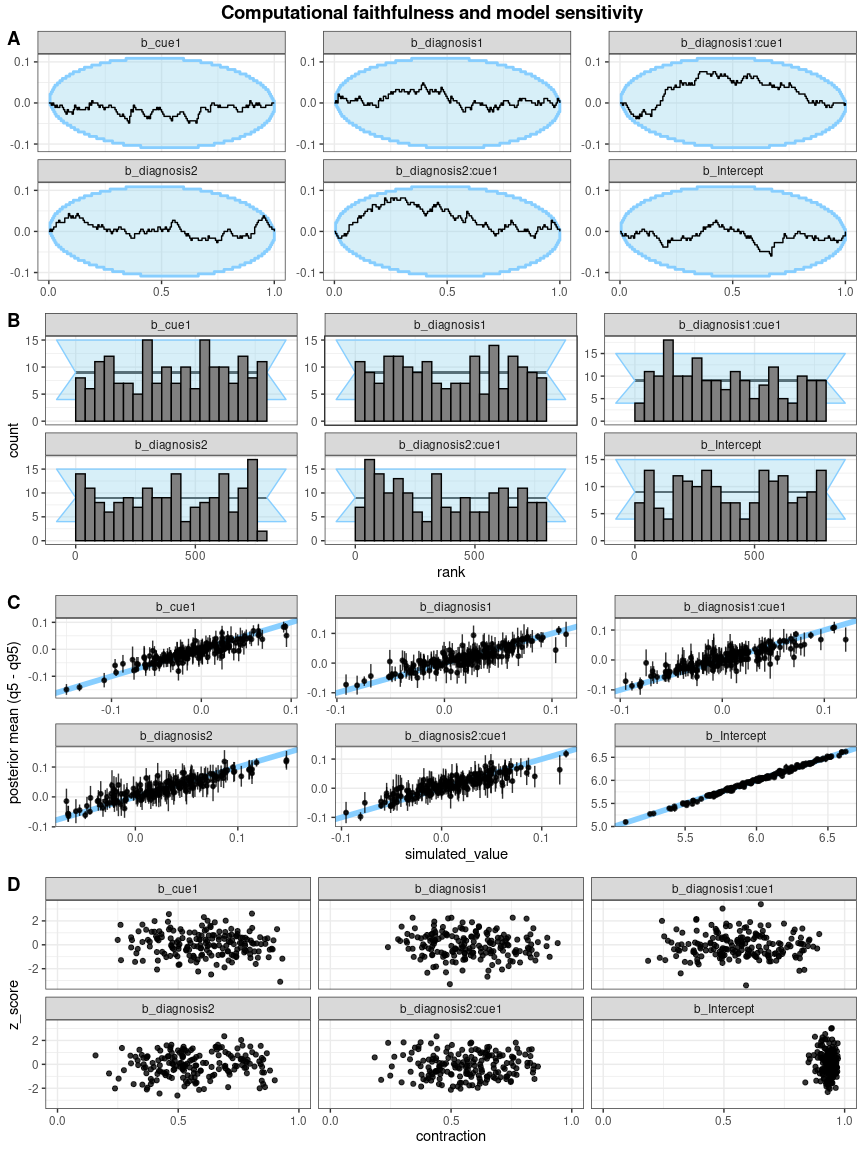
Next, we can plot the simulated values to perform prior predictive checks.

# create a matrix out of generated data  
dvname = gsub(" ", "", gsub("[\\|~].\*", "", f.fab)[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  
  
# set large values to a max  
dvfakemat[dvfakemat > 2000] = 2000  
  
# compute one histogram per simulated data-set   
binwidth = 20   
breaks = seq(0, max(dvfakemat, na.rm=T) + binwidth, binwidth)   
histmat = matrix(NA, ncol = length(dat), nrow = length(breaks)-1)   
for (i in 1:nrow(truePars)) {  
 histmat[,i] = hist(dvfakemat[,i], breaks = breaks, plot = F)$counts   
}  
# for each bin, compute quantiles across histograms   
probs = seq(0.1, 0.9, 0.1)   
quantmat= as.data.frame(matrix(NA, nrow=dim(histmat)[1], ncol = length(probs)))  
names(quantmat) = paste0("p", probs)  
for (i in 1:dim(histmat)[1]) {  
 quantmat[i,] = quantile(histmat[i,], p = probs)  
}  
quantmat$x = breaks[2:length(breaks)] - binwidth/2 # add bin mean   
p1 = 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) +   
 labs(title = "Distribution of simulated discriminations", y = "", x = "") +  
 theme\_bw()  
  
tmpM = apply(dvfakemat, 2, mean) # mean   
tmpSD = apply(dvfakemat, 2, sd)   
p2 = ggplot() +   
 stat\_bin(aes(x = tmpM), fill = c\_dark) +   
 labs(x = "Mean RTs (ms)", title = "Means of simulated RTs") +  
 theme\_bw()  
p3 = ggplot() +   
 stat\_bin(aes(x = tmpSD), fill = c\_dark) +   
 labs(x = "SD RTs (ms)", title = "SDs of simulated RTs") +  
 theme\_bw()  
p = ggarrange(p1,   
 ggarrange(p2, p3, ncol = 2, labels = c("B", "C")),   
 nrow = 2, labels = "A")  
annotate\_figure(p,   
 top = text\_grob("Prior predictive checks: reaction times",   
 face = "bold", size = 14))



Subfigure A shows the distribution of the simulated data with bluer bands being more likely than greyer bands. It shows a distribution that fits our expectations about reaction times in a simple decision task. The same applies to the distribution of the means and standard deviations in the simulated datasets. We go ahead with these priors and check the results of the SBC. We only plot the results from the models that had no divergence issues.

# 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$sim\_id))  
p1 = plot\_ecdf\_diff(df.results.b) + theme\_bw() + theme(legend.position = "none") +  
 scale\_x\_continuous(breaks=scales::pretty\_breaks(n = 3)) +  
 scale\_y\_continuous(breaks=scales::pretty\_breaks(n = 3))  
p2 = plot\_rank\_hist(df.results.b, bins = 20) + theme\_bw() +  
 scale\_x\_continuous(breaks=scales::pretty\_breaks(n = 3)) +  
 scale\_y\_continuous(breaks=scales::pretty\_breaks(n = 3))  
p3 = plot\_sim\_estimated(df.results.b, alpha = .8) + theme\_bw() +  
 scale\_x\_continuous(breaks=scales::pretty\_breaks(n = 3)) +  
 scale\_y\_continuous(breaks=scales::pretty\_breaks(n = 3))  
p4 = plot\_contraction(df.results.b,   
 prior\_sd = setNames(  
 c(as.numeric(  
 gsub(".\*, (.+)\\).\*", "\\1",   
 priors[priors$class == "Intercept",]$prior)),   
 as.numeric(  
 gsub(".\*, (.+)\\).\*", "\\1",   
 priors[priors$class == "b",]$prior))),   
 unique(df.results.b$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))



Next, 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.

Then, 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). All of this looks good for this model.

### Posterior predictive checks

As the next step, we fit the model to the data, check whether there are divergence or rhat issues, and then check whether the chains have converged.

# fit the full model  
set.seed(2469)  
m.fab = brm(f.fab,  
 df.fab, prior = priors,  
 family = shifted\_lognormal,  
 iter = iter, warmup = warm,  
 backend = "cmdstanr", threads = threading(8),  
 file = "m\_fab\_full"  
 )  
rstan::check\_hmc\_diagnostics(m.fab$fit)

##   
## Divergences:

## 0 of 8000 iterations ended with a divergence.

##   
## Tree depth:

## 0 of 8000 iterations saturated the maximum tree depth of 10.

##   
## Energy:

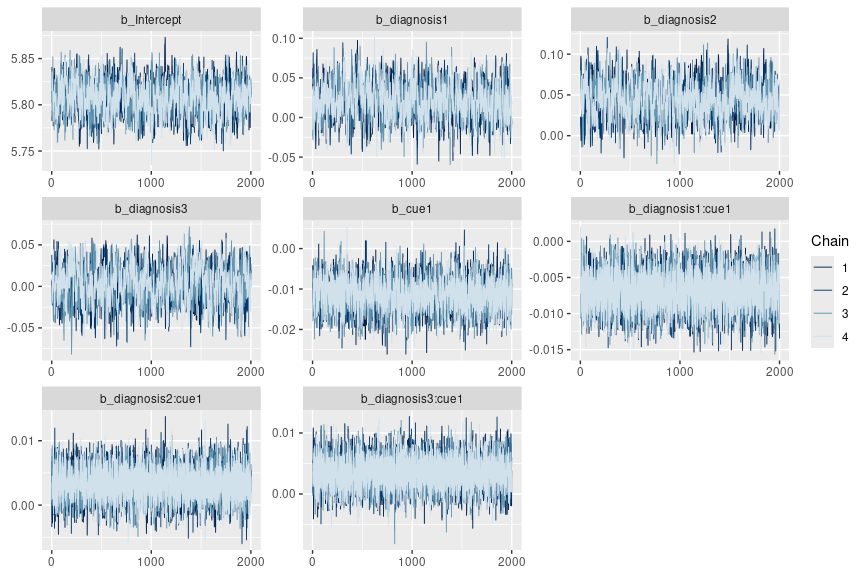
## E-BFMI indicated no pathological behavior.

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

## [1] 0

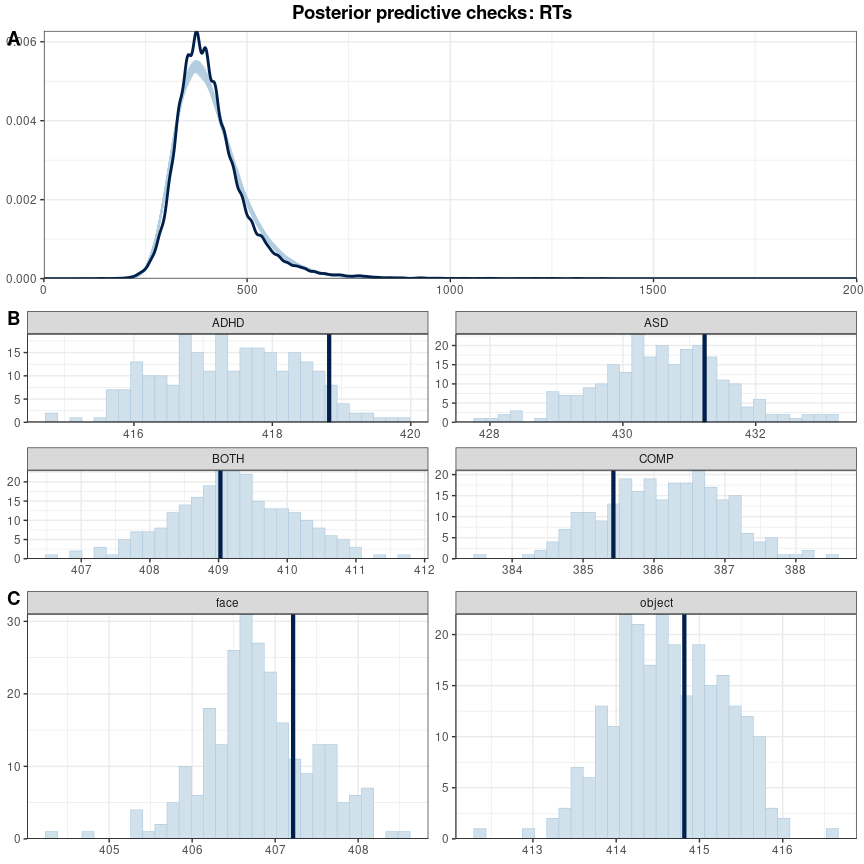
# check the trace plots  
post.draws = as\_draws\_df(m.fab)  
mcmc\_trace(post.draws, regex\_pars = "^b\_",  
 facet\_args = list(ncol = 3)) +  
 scale\_x\_continuous(breaks=scales::pretty\_breaks(n = 3)) +  
 scale\_y\_continuous(breaks=scales::pretty\_breaks(n = 3))

## Scale for x is already present.  
## Adding another scale for x, which will replace the existing scale.



This model has no pathological behaviour with E-BFMI, 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 posterior predictions  
post.pred = posterior\_predict(m.fab, ndraws = nsim)  
  
# check the fit of the predicted data compared to the real data  
p1 = pp\_check(m.fab, ndraws = nsim) +   
 theme\_bw() + theme(legend.position = "none") + xlim(0, 2000)  
  
# get rid of NAs in data frame for plotting  
df.fab.na = df.fab[!is.na(df.fab$rt.cor),]  
  
# distributions of means compared to the real values per group  
p2 = ppc\_stat\_grouped(df.fab.na$rt.cor, post.pred, df.fab.na$diagnosis) +   
 theme\_bw() + theme(legend.position = "none")  
  
# distributions of means compared to the real values per cue  
p3 = ppc\_stat\_grouped(df.fab.na$rt.cor, post.pred, df.fab.na$cue) +   
 theme\_bw() + theme(legend.position = "none")  
  
p = ggarrange(p1, p2, p3,  
 nrow = 3, ncol = 1, labels = "AUTO")  
annotate\_figure(p,   
 top = text\_grob("Posterior predictive checks: RTs",   
 face = "bold", size = 14))



Although the overall shape in subfigure A of the simulated data fits well with the real data, the model seems to underestimate the reaction times of the ADHD and ASD groups and overestimate the reaction times of the COMP group: the dark blue line shows the mean of the actual dataset while the light blue bars show the distribution of the predicted data.

Since we are interested in accurate estimates, we decide to aggregate with the median of the reaction times per stimulus (face-object cue combination) and cue. Then, there are no missing values in the data and we model an estimate for each specific stimulus and cue combination for each participant.

## Aggregated model

First, we compute the aggregation and have a quick look at the resulting data.

# keep full dataframe  
df.fab.full = df.fab  
  
# aggregate reaction times  
df.fab = df.fab %>%  
 group\_by(subID, diagnosis, stm, cue) %>%  
 summarise(  
 rt.cor = median(rt.cor, na.rm = T)  
 ) %>% ungroup() %>%  
 mutate\_if(is.character, as.factor)  
  
# set and print the contrasts  
contrasts(df.fab$cue) = contr.sum(2)  
contrasts(df.fab$cue)

## [,1]  
## face 1  
## object -1

contrasts(df.fab$diagnosis) = contr.sum(4)[c(1,2,4,3),]  
contrasts(df.fab$diagnosis)

## [,1] [,2] [,3]  
## ADHD 1 0 0  
## ASD 0 1 0  
## BOTH -1 -1 -1  
## COMP 0 0 1

summary(df.fab)

## subID diagnosis stm cue rt.cor   
## 1 : 72 ADHD:1656 1\_10 : 188 face :3384 Min. :256.0   
## 2 : 72 ASD :1728 1\_11 : 188 object:3384 1st Qu.:364.0   
## 3 : 72 BOTH:1656 1\_12 : 188 Median :394.8   
## 4 : 72 COMP:1728 1\_7 : 188 Mean :406.6   
## 5 : 72 1\_8 : 188 3rd Qu.:435.5   
## 6 : 72 1\_9 : 188 Max. :919.0   
## (Other):6336 (Other):5640

There are now no NAs in the data, because no one made an error on all instances of one stimulus combination.

### Stimulation-based calibration

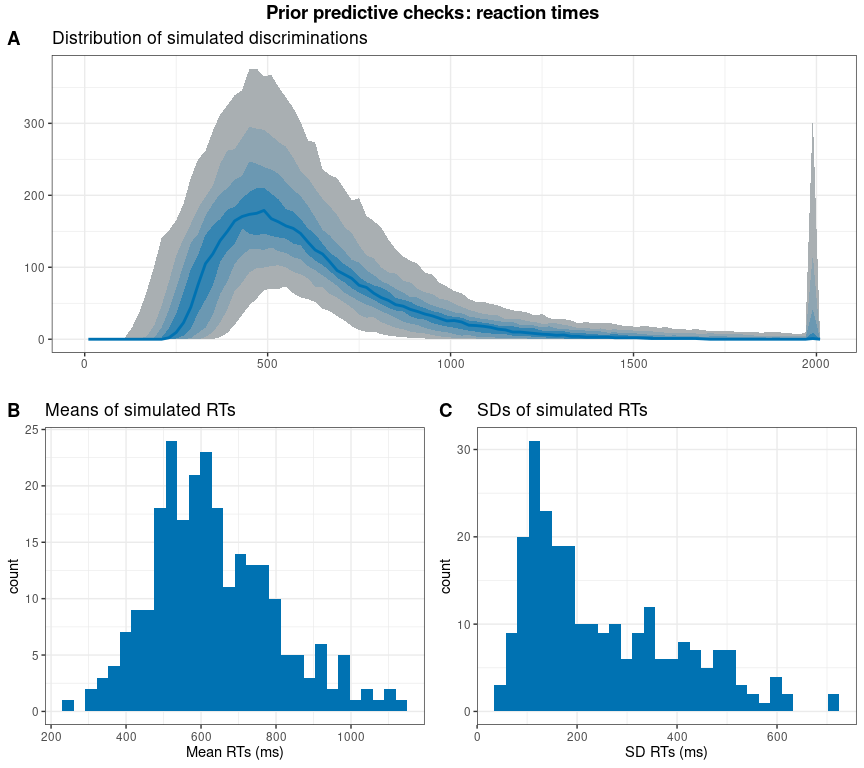
We again perform an SBC. The model formula and priors can stay the same.

code = "FAB\_agg"  
  
# check if the SBC already exists  
if (file.exists(file.path(cache\_dir, sprintf("df\_res\_%s.rds", code)))) {  
 # load in the results of the SBC  
 df.results = readRDS(file.path(cache\_dir, sprintf("df\_res\_%s.rds", code)))  
 df.backend = readRDS(file.path(cache\_dir, sprintf("df\_div\_%s.rds", code)))  
 dat = readRDS(file.path(cache\_dir, sprintf("dat\_%s.rds", code)))  
} else {  
 # perform the SBC  
 gen = SBC\_generator\_brms(f.fab, data = df.fab, prior = priors,   
 family = shifted\_lognormal,  
 thin = 50, warmup = 20000, refresh = 2000)  
 bck = SBC\_backend\_brms\_from\_generator(gen, chains = 4, thin = 1,  
 init = 0.1, warmup = warm, iter = iter)  
 set.seed(468)  
 if (file.exists(file.path(cache\_dir, sprintf("dat\_%s.rds", code)))) {  
 dat = readRDS(file.path(cache\_dir, sprintf("dat\_%s.rds", code)))  
 } else {  
 dat = generate\_datasets(gen, nsim)  
 saveRDS(dat, file.path(cache\_dir, sprintf("dat\_%s.rds", code)))  
 }  
 res = compute\_SBC(dat,   
 bck,  
 cache\_mode = "results",   
 cache\_location = file.path(cache\_dir, sprintf("res\_%s", code)))  
 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")))  
}  
  
set.seed(4682)

We start by investigating the rhats and the number of divergent samples. This shows that 3 of 250 simulations had at least one parameter that had an rhat of at least 1.05, and 2 models had divergent samples (mean number of samples of the simulations with divergent samples: 17.5). This suggests that this model performs well enough and only few simulated models exhibit issues.

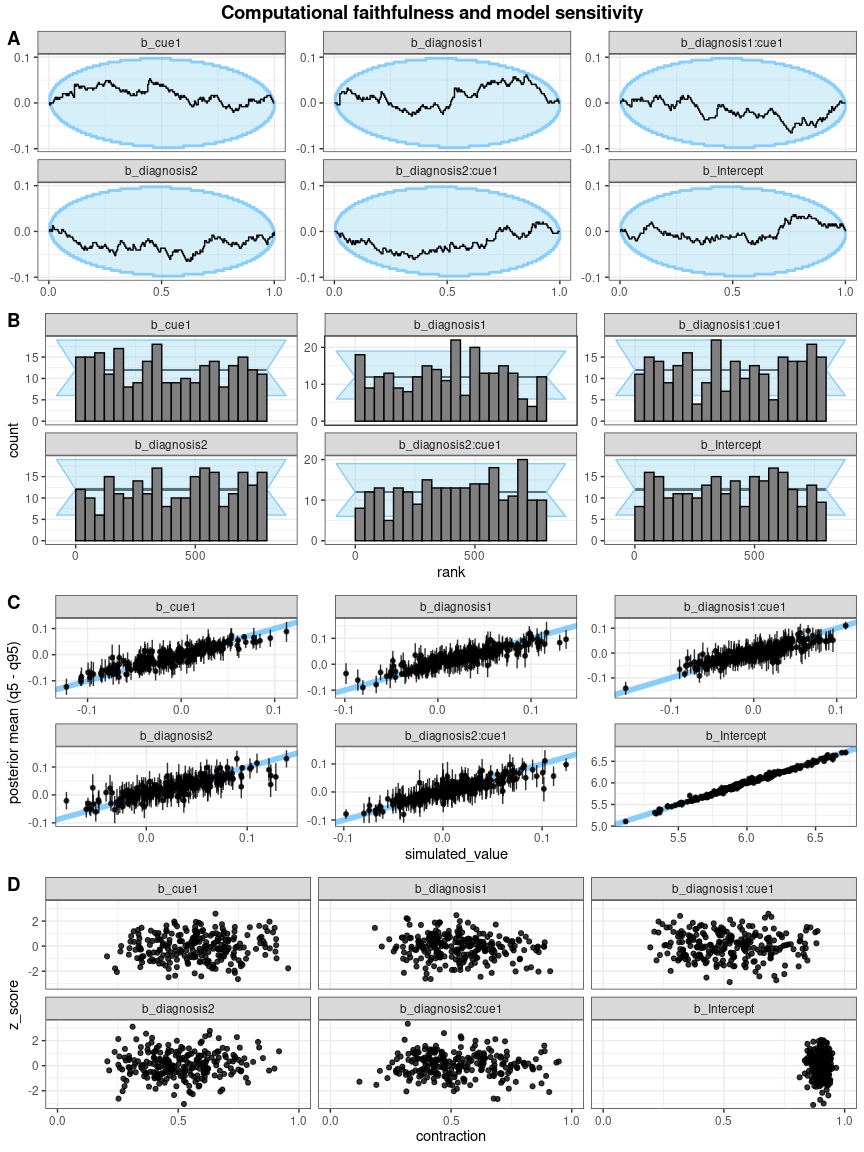
Next, we can plot the simulated values to perform prior predictive checks.

# create a matrix out of generated data  
dvname = gsub(" ", "", gsub("[\\|~].\*", "", f.fab)[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  
  
# set large values to a max  
dvfakemat[dvfakemat > 2000] = 2000  
  
# compute one histogram per simulated data-set   
binwidth = 20   
breaks = seq(0, max(dvfakemat, na.rm=T) + binwidth, binwidth)   
histmat = matrix(NA, ncol = length(dat), nrow = length(breaks)-1)   
for (i in 1:nrow(truePars)) {  
 histmat[,i] = hist(dvfakemat[,i], breaks = breaks, plot = F)$counts   
}  
# for each bin, compute quantiles across histograms   
probs = seq(0.1, 0.9, 0.1)   
quantmat= as.data.frame(matrix(NA, nrow=dim(histmat)[1], ncol = length(probs)))  
names(quantmat) = paste0("p", probs)  
for (i in 1:dim(histmat)[1]) {  
 quantmat[i,] = quantile(histmat[i,], p = probs)  
}  
quantmat$x = breaks[2:length(breaks)] - binwidth/2 # add bin mean   
p1 = 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) +   
 labs(title = "Distribution of simulated discriminations", y = "", x = "") +  
 theme\_bw()  
  
tmpM = apply(dvfakemat, 2, mean) # mean   
tmpSD = apply(dvfakemat, 2, sd)   
p2 = ggplot() +   
 stat\_bin(aes(x = tmpM), fill = c\_dark) +   
 labs(x = "Mean RTs (ms)", title = "Means of simulated RTs") +  
 theme\_bw()  
p3 = ggplot() +   
 stat\_bin(aes(x = tmpSD), fill = c\_dark) +   
 labs(x = "SD RTs (ms)", title = "SDs of simulated RTs") +  
 theme\_bw()  
p = ggarrange(p1,   
 ggarrange(p2, p3, ncol = 2, labels = c("B", "C")),   
 nrow = 2, labels = "A")  
annotate\_figure(p,   
 top = text\_grob("Prior predictive checks: reaction times",   
 face = "bold", size = 14))



Again, this all looks good.

# 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$sim\_id))  
p1 = plot\_ecdf\_diff(df.results.b) + theme\_bw() + theme(legend.position = "none") +  
 scale\_x\_continuous(breaks=scales::pretty\_breaks(n = 3)) +  
 scale\_y\_continuous(breaks=scales::pretty\_breaks(n = 3))  
p2 = plot\_rank\_hist(df.results.b, bins = 20) + theme\_bw() +  
 scale\_x\_continuous(breaks=scales::pretty\_breaks(n = 3)) +  
 scale\_y\_continuous(breaks=scales::pretty\_breaks(n = 3))  
p3 = plot\_sim\_estimated(df.results.b, alpha = .8) + theme\_bw() +  
 scale\_x\_continuous(breaks=scales::pretty\_breaks(n = 3)) +  
 scale\_y\_continuous(breaks=scales::pretty\_breaks(n = 3))  
p4 = plot\_contraction(df.results.b,   
 prior\_sd = setNames(  
 c(as.numeric(  
 gsub(".\*, (.+)\\).\*", "\\1",   
 priors[priors$class == "Intercept",]$prior)),   
 as.numeric(  
 gsub(".\*, (.+)\\).\*", "\\1",   
 priors[priors$class == "b",]$prior))),   
 unique(df.results.b$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))



Rank histogramms, sample ECDF, the relationship between the simulated true parameters and the posterior estimates as well as z-score and posterior contraction of our population-level predictors all are acceptable for this model as well.

### Posterior predictive checks

As the next step, we fit the model to the data, check whether there are divergence or rhat issues, and then check whether the chains have converged.

# fit the final model  
set.seed(6824)  
m.fab = brm(f.fab,  
 df.fab, prior = priors,  
 family = shifted\_lognormal,  
 iter = iter, warmup = warm,  
 backend = "cmdstanr", threads = threading(8),  
 file = "m\_fab\_final"  
 )  
rstan::check\_hmc\_diagnostics(m.fab$fit)

##   
## Divergences:

## 0 of 8000 iterations ended with a divergence.

##   
## Tree depth:

## 0 of 8000 iterations saturated the maximum tree depth of 10.

##   
## Energy:

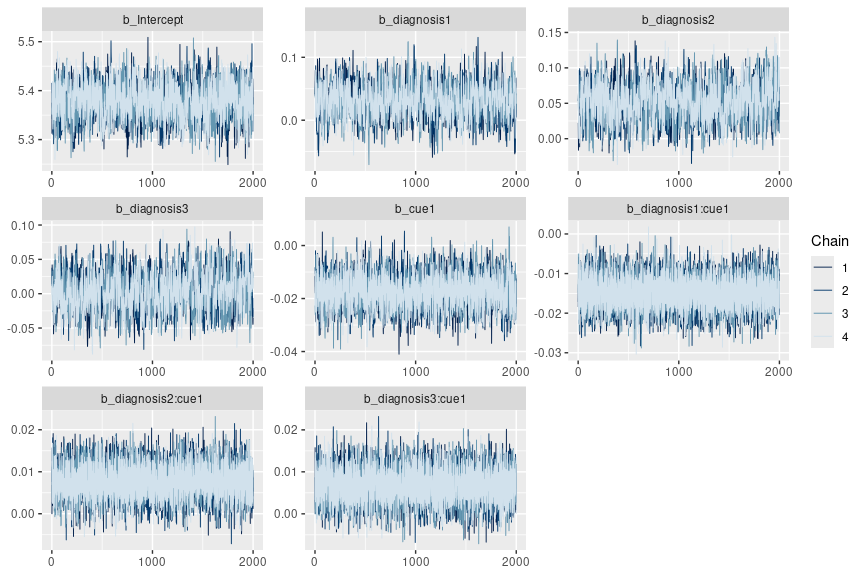
## E-BFMI indicated no pathological behavior.

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

## [1] 0

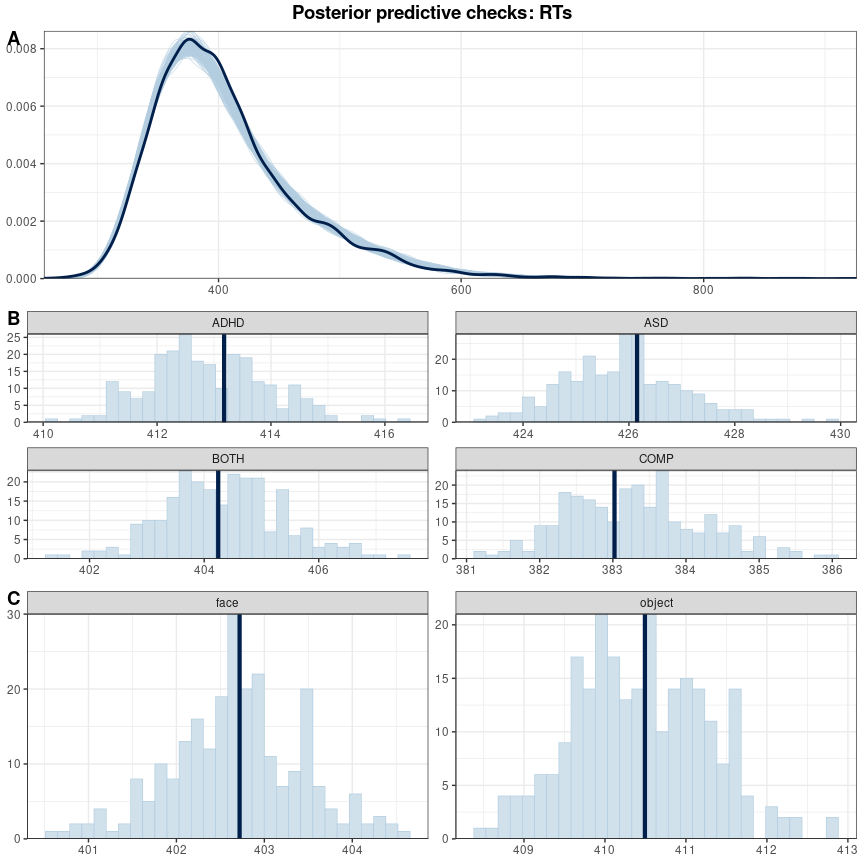
# check the trace plots  
post.draws = as\_draws\_df(m.fab)  
mcmc\_trace(post.draws, regex\_pars = "^b\_",  
 facet\_args = list(ncol = 3)) +  
 scale\_x\_continuous(breaks=scales::pretty\_breaks(n = 3)) +  
 scale\_y\_continuous(breaks=scales::pretty\_breaks(n = 3))

## Scale for x is already present.  
## Adding another scale for x, which will replace the existing scale.



This model has no pathological behaviour with E-BFMI, 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 posterior predictions  
post.pred = posterior\_predict(m.fab, ndraws = nsim)  
  
# check the fit of the predicted data compared to the real data  
p1 = pp\_check(m.fab, ndraws = nsim) +   
 theme\_bw() + theme(legend.position = "none")  
  
# distributions of means compared to the real values per group  
p2 = ppc\_stat\_grouped(df.fab$rt.cor, post.pred, df.fab$diagnosis) +   
 theme\_bw() + theme(legend.position = "none")  
  
# distributions of means compared to the real values per cue  
p3 = ppc\_stat\_grouped(df.fab$rt.cor, post.pred, df.fab$cue) +   
 theme\_bw() + theme(legend.position = "none")  
  
p = ggarrange(p1, p2, p3,  
 nrow = 3, ncol = 1, labels = "AUTO")  
annotate\_figure(p,   
 top = text\_grob("Posterior predictive checks: RTs",   
 face = "bold", size = 14))



This model fits our data much better.

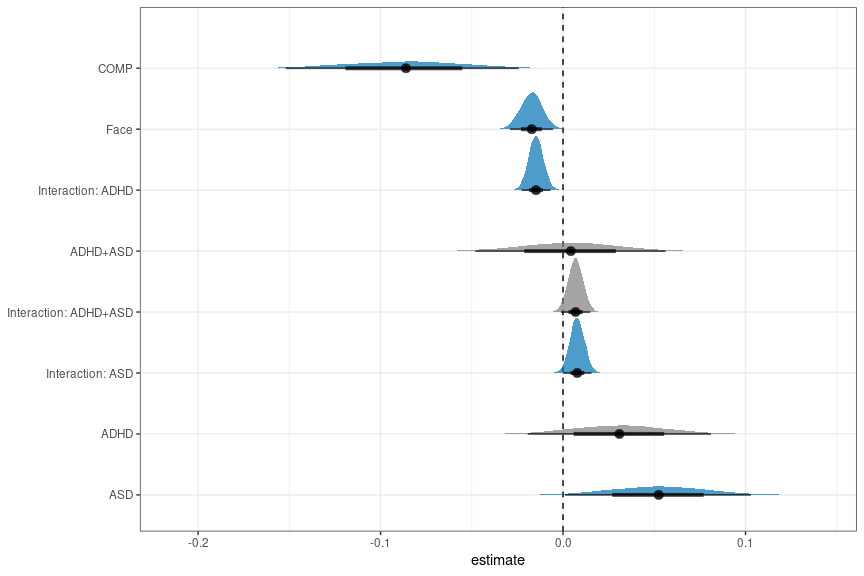
### Inferences

Now that we are convinced that we can trust our model, we have a look at its estimate and use the hypothesis function to assess our hypotheses and perform explorative tests.

# print a summary  
summary(m.fab)

## Family: shifted\_lognormal   
## Links: mu = identity; sigma = identity; ndt = identity   
## Formula: rt.cor ~ diagnosis \* cue + (cue | subID) + (cue \* diagnosis | stm)   
## Data: df.fab (Number of observations: 6768)   
## Draws: 4 chains, each with iter = 3000; warmup = 1000; thin = 1;  
## total post-warmup draws = 8000  
##   
## Multilevel Hyperparameters:  
## ~stm (Number of levels: 36)   
## Estimate Est.Error l-95% CI u-95% CI Rhat  
## sd(Intercept) 0.03 0.00 0.02 0.03 1.00  
## sd(cue1) 0.03 0.00 0.03 0.04 1.00  
## sd(diagnosis1) 0.01 0.00 0.00 0.02 1.00  
## sd(diagnosis2) 0.00 0.00 0.00 0.01 1.00  
## sd(diagnosis3) 0.01 0.00 0.00 0.01 1.00  
## sd(cue1:diagnosis1) 0.00 0.00 0.00 0.01 1.00  
## sd(cue1:diagnosis2) 0.00 0.00 0.00 0.01 1.00  
## sd(cue1:diagnosis3) 0.00 0.00 0.00 0.01 1.00  
## cor(Intercept,cue1) -0.30 0.15 -0.57 0.03 1.00  
## cor(Intercept,diagnosis1) -0.12 0.26 -0.60 0.45 1.00  
## cor(cue1,diagnosis1) 0.01 0.26 -0.50 0.51 1.00  
## cor(Intercept,diagnosis2) -0.07 0.29 -0.60 0.51 1.00  
## cor(cue1,diagnosis2) -0.05 0.29 -0.58 0.51 1.00  
## cor(diagnosis1,diagnosis2) -0.01 0.30 -0.58 0.57 1.00  
## cor(Intercept,diagnosis3) 0.06 0.27 -0.49 0.57 1.00  
## cor(cue1,diagnosis3) 0.18 0.27 -0.41 0.66 1.00  
## cor(diagnosis1,diagnosis3) -0.12 0.30 -0.66 0.49 1.00  
## cor(diagnosis2,diagnosis3) -0.08 0.31 -0.65 0.54 1.00  
## cor(Intercept,cue1:diagnosis1) 0.03 0.28 -0.51 0.55 1.00  
## cor(cue1,cue1:diagnosis1) -0.06 0.28 -0.58 0.49 1.00  
## cor(diagnosis1,cue1:diagnosis1) 0.00 0.30 -0.56 0.57 1.00  
## cor(diagnosis2,cue1:diagnosis1) -0.00 0.30 -0.57 0.56 1.00  
## cor(diagnosis3,cue1:diagnosis1) -0.03 0.30 -0.59 0.54 1.00  
## cor(Intercept,cue1:diagnosis2) -0.21 0.28 -0.69 0.40 1.00  
## cor(cue1,cue1:diagnosis2) -0.01 0.28 -0.54 0.53 1.00  
## cor(diagnosis1,cue1:diagnosis2) 0.03 0.29 -0.54 0.58 1.00  
## cor(diagnosis2,cue1:diagnosis2) 0.04 0.30 -0.54 0.61 1.00  
## cor(diagnosis3,cue1:diagnosis2) -0.05 0.29 -0.59 0.53 1.00  
## cor(cue1:diagnosis1,cue1:diagnosis2) -0.09 0.31 -0.65 0.53 1.00  
## cor(Intercept,cue1:diagnosis3) -0.10 0.28 -0.62 0.46 1.00  
## cor(cue1,cue1:diagnosis3) -0.15 0.28 -0.65 0.43 1.00  
## cor(diagnosis1,cue1:diagnosis3) 0.05 0.30 -0.52 0.61 1.00  
## cor(diagnosis2,cue1:diagnosis3) 0.06 0.30 -0.53 0.62 1.00  
## cor(diagnosis3,cue1:diagnosis3) -0.02 0.30 -0.59 0.55 1.00  
## cor(cue1:diagnosis1,cue1:diagnosis3) -0.08 0.30 -0.63 0.52 1.00  
## cor(cue1:diagnosis2,cue1:diagnosis3) -0.01 0.30 -0.58 0.57 1.00  
## Bulk\_ESS Tail\_ESS  
## sd(Intercept) 2578 4611  
## sd(cue1) 2693 4533  
## sd(diagnosis1) 2411 3394  
## sd(diagnosis2) 4155 4612  
## sd(diagnosis3) 3402 4152  
## sd(cue1:diagnosis1) 3823 4565  
## sd(cue1:diagnosis2) 3982 4324  
## sd(cue1:diagnosis3) 3597 3381  
## cor(Intercept,cue1) 1826 3831  
## cor(Intercept,diagnosis1) 13054 5814  
## cor(cue1,diagnosis1) 13349 6130  
## cor(Intercept,diagnosis2) 16827 5305  
## cor(cue1,diagnosis2) 17348 5809  
## cor(diagnosis1,diagnosis2) 9760 6457  
## cor(Intercept,diagnosis3) 15806 5807  
## cor(cue1,diagnosis3) 12478 6278  
## cor(diagnosis1,diagnosis3) 7439 6368  
## cor(diagnosis2,diagnosis3) 7729 6017  
## cor(Intercept,cue1:diagnosis1) 15320 6384  
## cor(cue1,cue1:diagnosis1) 16067 5911  
## cor(diagnosis1,cue1:diagnosis1) 10215 6390  
## cor(diagnosis2,cue1:diagnosis1) 7492 6359  
## cor(diagnosis3,cue1:diagnosis1) 7425 7042  
## cor(Intercept,cue1:diagnosis2) 11799 6187  
## cor(cue1,cue1:diagnosis2) 15752 5325  
## cor(diagnosis1,cue1:diagnosis2) 10817 6320  
## cor(diagnosis2,cue1:diagnosis2) 8245 6495  
## cor(diagnosis3,cue1:diagnosis2) 6974 6717  
## cor(cue1:diagnosis1,cue1:diagnosis2) 6068 6426  
## cor(Intercept,cue1:diagnosis3) 14585 6447  
## cor(cue1,cue1:diagnosis3) 11674 6152  
## cor(diagnosis1,cue1:diagnosis3) 8962 5900  
## cor(diagnosis2,cue1:diagnosis3) 6889 6461  
## cor(diagnosis3,cue1:diagnosis3) 8014 7067  
## cor(cue1:diagnosis1,cue1:diagnosis3) 6829 6746  
## cor(cue1:diagnosis2,cue1:diagnosis3) 6471 6914  
##   
## ~subID (Number of levels: 94)   
## Estimate Est.Error l-95% CI u-95% CI Rhat Bulk\_ESS Tail\_ESS  
## sd(Intercept) 0.21 0.02 0.18 0.24 1.00 1249 2259  
## sd(cue1) 0.02 0.00 0.01 0.02 1.00 3696 5257  
## cor(Intercept,cue1) -0.04 0.14 -0.32 0.24 1.00 7658 6980  
##   
## Regression Coefficients:  
## Estimate Est.Error l-95% CI u-95% CI Rhat Bulk\_ESS Tail\_ESS  
## Intercept 5.38 0.03 5.31 5.45 1.00 1000 1967  
## diagnosis1 0.03 0.03 -0.02 0.08 1.00 1053 1809  
## diagnosis2 0.05 0.03 0.00 0.10 1.00 959 1959  
## diagnosis3 0.00 0.03 -0.05 0.06 1.00 1055 2006  
## cue1 -0.02 0.01 -0.03 -0.01 1.00 2074 3747  
## diagnosis1:cue1 -0.01 0.00 -0.02 -0.01 1.00 8305 7001  
## diagnosis2:cue1 0.01 0.00 0.00 0.02 1.00 8591 7066  
## diagnosis3:cue1 0.01 0.00 -0.00 0.01 1.00 8207 7062  
##   
## Further Distributional Parameters:  
## Estimate Est.Error l-95% CI u-95% CI Rhat Bulk\_ESS Tail\_ESS  
## sigma 0.13 0.00 0.13 0.14 1.00 5963 5842  
## ndt 183.19 5.64 171.94 193.62 1.00 5982 5698  
##   
## 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).

# get the estimates and compute groups  
df.m.fab = as\_draws\_df(m.fab) %>%   
 select(starts\_with("b\_")) %>%  
 mutate(  
 b\_COMP = - b\_diagnosis1 - b\_diagnosis2 - b\_diagnosis3,  
 ASD = b\_Intercept + b\_diagnosis2,  
 ADHD = b\_Intercept + b\_diagnosis1,  
 BOTH = b\_Intercept + b\_diagnosis3,  
 COMP = b\_Intercept + b\_COMP  
 )  
  
# plot the posterior distributions  
df.m.fab %>%   
 select(starts\_with("b\_")) %>%  
 pivot\_longer(cols = starts\_with("b\_"), names\_to = "coef", values\_to = "estimate") %>%  
 filter(coef != "b\_Intercept") %>%  
 mutate(  
 coef = case\_match(coef,  
 "b\_diagnosis1" ~ "ADHD",  
 "b\_diagnosis2" ~ "ASD",  
 "b\_diagnosis3" ~ "ADHD+ASD",  
 "b\_COMP" ~ "COMP",  
 "b\_cue1" ~ "Face",  
 "b\_diagnosis1:cue1" ~ "Interaction: ADHD",  
 "b\_diagnosis2:cue1" ~ "Interaction: ASD",  
 "b\_diagnosis3:cue1" ~ "Interaction: ADHD+ASD"  
 ),  
 coef = fct\_reorder(coef, desc(estimate))  
 ) %>%   
 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")



# H1a: FAB effect in COMP  
h1a = hypothesis(m.fab,   
 "0 < 2\*(diagnosis1:cue1 + diagnosis2:cue1 + diagnosis3:cue1 - cue1)")  
h1a

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(2\*(diagnosis... < 0 -0.03 0.02 -0.06 -0.01 77.43  
## Post.Prob Star  
## 1 0.99 \*  
## ---  
## '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.

# H1b: ADHD slower than COMP  
h1b = hypothesis(m.fab,   
 "0 < 2\*diagnosis1 + diagnosis2 + diagnosis3")  
h1b

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(2\*diagnosis1... < 0 -0.12 0.05 -0.2 -0.04 139.35  
## Post.Prob Star  
## 1 0.99 \*  
## ---  
## '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.

# H1c: ASD slower than COMP  
h1c = hypothesis(m.fab,   
 "0 < 2\*diagnosis2 + diagnosis1 + diagnosis3")  
h1c

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(2\*diagnosis2... < 0 -0.14 0.05 -0.22 -0.06 443.44  
## Post.Prob Star  
## 1 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.

# H1d: FAB in ASD decreased compared to COMP  
h1d = hypothesis(m.fab,   
 "0 < 4\*diagnosis2:cue1 + 2\*diagnosis1:cue1 + 2\*diagnosis3:cue1")  
h1d

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(4\*diagnosis2... < 0 -0.02 0.01 -0.04 0.01 6.9  
## Post.Prob Star  
## 1 0.87   
## ---  
## '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.

# H1e: FAB in ADHD differs from FAB in COMP (undirected)  
h1e = hypothesis(m.fab,   
 "0 > 4\*diagnosis1:cue1 + 2\*diagnosis2:cue1 + 2\*diagnosis3:cue1",   
 alpha = 0.025)  
h1e

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(4\*diagnosis1... > 0 0.03 0.01 0 0.06 71.73  
## Post.Prob Star  
## 1 0.99 \*  
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

# Exploration  
  
# E1: FAB generally  
e1 = hypothesis(m.fab, "2\*cue1 < 0", alpha = 0.025)  
e1

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio Post.Prob Star  
## 1 (2\*cue1) < 0 -0.03 0.01 -0.06 -0.01 306.69 1 \*  
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

# E2: FAB effect in ADHD  
e2 = hypothesis(m.fab, "0 < -2\*cue1 - 2\*diagnosis1:cue1", alpha = 0.025)  
e2

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(-2\*cue1-2\*di... < 0 -0.06 0.01 -0.09 -0.04 Inf  
## Post.Prob Star  
## 1 1 \*  
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

# E3: FAB effect in ASD  
e3 = hypothesis(m.fab, "0 < -2\*cue1 - 2\*diagnosis2:cue1", alpha = 0.025)  
e3

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(-2\*cue1-2\*di... < 0 -0.02 0.01 -0.05 0.01 9.39  
## Post.Prob Star  
## 1 0.9   
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

# E4: FAB effect in ADHD+ASD  
e4 = hypothesis(m.fab, "0 < -2\*cue1 - 2\*diagnosis3:cue1", alpha = 0.025)  
e4

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(-2\*cue1-2\*di... < 0 -0.02 0.01 -0.05 0.01 12.84  
## Post.Prob Star  
## 1 0.93   
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

# E5: FAB in ADHD differs from FAB in ASD  
e5 = hypothesis(m.fab,   
 "0 < -2\*diagnosis1:cue1 + 2\*diagnosis2:cue1", alpha = 0.025)  
e5

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(-2\*diagnosis... < 0 -0.05 0.01 -0.07 -0.02 7999  
## Post.Prob Star  
## 1 1 \*  
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

# E6: FAB in ADHD differs from FAB in BOTH  
e6 = hypothesis(m.fab,   
 "0 < -2\*diagnosis1:cue1 + 2\*diagnosis3:cue1", alpha = 0.025)  
e6

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(-2\*diagnosis... < 0 -0.04 0.01 -0.07 -0.02 2665.67  
## Post.Prob Star  
## 1 1 \*  
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

# E7: FAB in ASD differs from FAB in BOTH  
e7 = hypothesis(m.fab,   
 "0 < -2\*diagnosis2:cue1 + 2\*diagnosis3:cue1", alpha = 0.025)  
e7

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(-2\*diagnosis... < 0 0 0.01 -0.02 0.03 0.79  
## Post.Prob Star  
## 1 0.44   
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

# E8: FAB in COMP differs from FAB in BOTH  
e8 = hypothesis(m.fab,   
 "0 < 2\*diagnosis1:cue1 + 2\*diagnosis2:cue1 + 4\*diagnosis3:cue1",   
 alpha = 0.025)  
e8

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(2\*diagnosis1... < 0 -0.01 0.01 -0.04 0.01 5.26  
## Post.Prob Star  
## 1 0.84   
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

# E9: face in COMP versus face in ADHD  
e9 = hypothesis(m.fab,   
 "0 < 2\*diagnosis1 + diagnosis2 + diagnosis3 +  
 2\*diagnosis1:cue1 + diagnosis2:cue1 + diagnosis3:cue1",   
 alpha = 0.025)  
e9

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(2\*diagnosis1... < 0 -0.1 0.05 -0.2 -0.01 59.15  
## Post.Prob Star  
## 1 0.98 \*  
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

# E10: object in COMP versus object in ADHD  
e10 = hypothesis(m.fab,   
 "0 < 2\*diagnosis1 + diagnosis2 + diagnosis3 -  
 (2\*diagnosis1:cue1 + diagnosis2:cue1 + diagnosis3:cue1)",   
 alpha = 0.025)  
e10

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(2\*diagnosis1... < 0 -0.13 0.05 -0.23 -0.04 362.64  
## Post.Prob Star  
## 1 1 \*  
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

# E11: E10 > E9  
e11 = hypothesis(m.fab,   
 "2\*diagnosis1 + diagnosis2 + diagnosis3 -  
 (2\*diagnosis1:cue1 + diagnosis2:cue1 + diagnosis3:cue1) >  
 2\*diagnosis1 + diagnosis2 + diagnosis3 +  
 (2\*diagnosis1:cue1 + diagnosis2:cue1 + diagnosis3:cue1)",   
 alpha = 0.025)  
e11

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (2\*diagnosis1+dia... > 0 0.03 0.01 0 0.06 71.73  
## Post.Prob Star  
## 1 0.99 \*  
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

# extract predicted differences in ms instead of log data  
df.new = df.fab %>%   
 select(diagnosis, cue) %>%   
 distinct() %>%  
 mutate(  
 condition = paste(diagnosis, cue, sep = "\_")  
 )  
df.ms = as.data.frame(  
 fitted(m.fab, summary = F,   
 newdata = df.new %>% select(diagnosis, cue),   
 re\_formula = NA))  
colnames(df.ms) = df.new$condition  
  
st(df.ms,   
 summ = c('mean(x)','sd(x)','min(x)','pctile(x)[2.5]',  
 'pctile(x)[97.5]','max(x)'))

Summary Statistics

| Variable | Mean | Sd | Min | Pctile[2.5] | Pctile[97.5] | Max |
| --- | --- | --- | --- | --- | --- | --- |
| ADHD\_face | 402 | 7.4 | 369 | 387 | 416 | 434 |
| ADHD\_object | 416 | 8.2 | 387 | 400 | 432 | 448 |
| COMP\_face | 380 | 8 | 351 | 364 | 395 | 412 |
| COMP\_object | 387 | 8.3 | 358 | 370 | 403 | 422 |
| ASD\_face | 411 | 7.9 | 385 | 396 | 426 | 441 |
| ASD\_object | 416 | 8.2 | 389 | 400 | 432 | 445 |
| BOTH\_face | 400 | 7.6 | 375 | 386 | 415 | 432 |
| BOTH\_object | 405 | 7.9 | 378 | 389 | 420 | 437 |

# calculate our difference columns  
df.ms = df.ms %>%  
 mutate(  
 COMP = rowMeans(select(., matches("COMP\_.\*")), na.rm = T),  
 ADHD = rowMeans(select(., matches("ADHD\_.\*")), na.rm = T),  
 ASD = rowMeans(select(., matches("ASD\_.\*")), na.rm = T),  
 BOTH = rowMeans(select(., matches("BOTH\_.\*")), na.rm = T),  
 FAB = rowMeans(select(., matches(".\*\_object")), na.rm = T) -  
 rowMeans(select(., matches(".\*\_face")), na.rm = T),  
 FAB\_COMP = COMP\_object - COMP\_face,  
 FAB\_ADHD = ADHD\_object - ADHD\_face,  
 FAB\_ASD = ASD\_object - ASD\_face,  
 FAB\_BOTH = BOTH\_object - BOTH\_face,  
 h1b = ADHD - COMP,  
 h1c = ASD - COMP,  
 h1d = FAB\_COMP - FAB\_ASD,  
 h1e = FAB\_ADHD - FAB\_COMP,  
 BOTH\_COMP = BOTH - COMP  
 )  
  
st(df.ms %>%   
 mutate(  
 face = rowMeans(select(., matches(".\*\_face")), na.rm = T),  
 object = rowMeans(select(., matches(".\*\_object")), na.rm = T),  
 ADHDvCOMP\_face = ADHD\_face - COMP\_face,  
 ADHDvCOMP\_object = ADHD\_object - COMP\_object,  
 diff\_ADHDvCOMP = ADHDvCOMP\_object - ADHDvCOMP\_face  
 ) %>% select(face, object, FAB, ADHDvCOMP\_face, ADHDvCOMP\_object, diff\_ADHDvCOMP),  
 summ = c('mean(x)','sd(x)','min(x)','pctile(x)[2.5]','pctile(x)[97.5]','max(x)'))

Summary Statistics

| Variable | Mean | Sd | Min | Pctile[2.5] | Pctile[97.5] | Max |
| --- | --- | --- | --- | --- | --- | --- |
| face | 398 | 4.9 | 379 | 388 | 408 | 418 |
| object | 406 | 5.2 | 387 | 395 | 416 | 427 |
| FAB | 7.6 | 2.7 | -2.9 | 2.1 | 13 | 17 |
| ADHDvCOMP\_face | 21 | 10 | -12 | 0.87 | 41 | 66 |
| ADHDvCOMP\_object | 29 | 11 | -8 | 6.9 | 50 | 76 |
| diff\_ADHDvCOMP | 7.6 | 3 | -4.9 | 1.6 | 13 | 20 |

As hypothesised, both autistic adults and adults with ADHD exhibited increased overall reaction times compared with the COMP group (COMP - ADHD: *estimate* = -0.12 [-0.2, -0.04], *posterior probability* = 99.29%; COMP - ASD: *estimate* = -0.14 [-0.22, -0.06], *posterior probability* = 99.78%). The model predicts that participants in the comparison group react 25.08ms [5.02, 45.75] faster than the participants in the ADHD group and 29.91ms [9.08, 50.83] faster than autistic participants. Additionally, the model predicts that the participants in the comparison group react 19.02ms [-1.55, 38.96] faster than adults in the ADHD+ASD group.

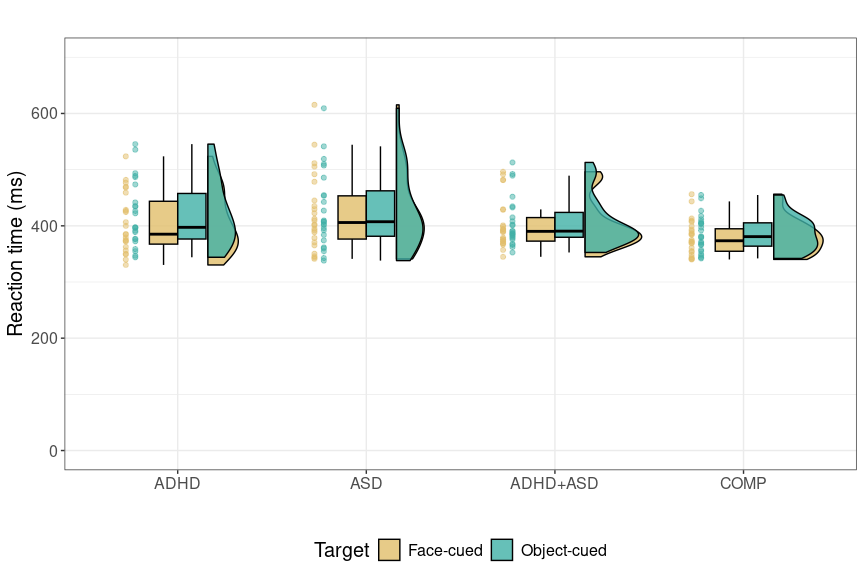
Our Bayesian linear mixed model with the median of correct reaction times as the outcome and diagnostic status, cue (face or object) and their interaction confirmed a face attention bias in our comparison group: COMP participants reacted faster in response to targets appearing on the side of the face compared to targets appearing on the side of the object (*estimate* = -0.03 [-0.06, -0.01], *posterior probability* = 98.72%). FAB was not credibly decreased in ASD participants compared to COMP participants (*estimate* = -0.02 [-0.04, 0.01], *posterior probability* = 87.34%). However, FAB was credibly higher in the ADHD than the COMP group (*estimate* = 0.03 [0, 0.06], *posterior probability* = 98.62%). Specifically, predicted reaction times based on the model estimate a FAB of 6.88ms [1.07, 12.92] in the COMP group, 14.47ms [8.08, 21.21] in the ADHD group, 4.37ms [-2.19, 10.97] in the ASD group as well as 4.57ms [-1.62, 10.71] in the ADHD+ASD group. These estimates are reflected in our exploration of FAB over all groups (*estimate* = -0.03 [-0.06, -0.01], *posterior probability* = 99.67%) as well as in the separate clinical groups with our model revealing a credible FAB effect in the ADHD (*estimate* = -0.06 [-0.09, -0.04], *posterior probability* = 100%) but not the ASD (*estimate* = -0.02 [-0.05, 0.01], *posterior probability* = 90.38%) and the ADHD+ASD group (*estimate* = -0.02 [-0.05, 0.01], *posterior probability* = 92.77%).

Exploration regarding comparison between ADHD and COMP on face and object separately:

* Face: *estimate* = -0.1 [-0.2, -0.01], *posterior probability* = 98.34%
* Object: *estimate* = -0.13 [-0.23, -0.04], *posterior probability* = 99.72%
* Object(ADHD-COMP) > Face(ADHD-COMP): *estimate* = 0.03 [0, 0.06], *posterior probability* = 98.62%

## Plots

# overall median reaction times  
df.fab %>%   
 group\_by(subID, diagnosis, cue) %>%  
 summarise(  
 rt.cor = mean(rt.cor, na.rm = T)  
 ) %>%  
 mutate(  
 diagnosis = recode(diagnosis, "BOTH" = "ADHD+ASD"),  
 Target = recode(cue, "face" = "Face-cued", "object" = "Object-cued")  
 ) %>%   
 ggplot(aes(diagnosis, rt.cor, fill = Target, colour = Target)) + #  
 geom\_rain(rain.side = 'r',  
boxplot.args = list(color = "black", outlier.shape = NA, show\_guide = FALSE, alpha = .8),  
violin.args = list(color = "black", outlier.shape = NA, alpha = .8),  
boxplot.args.pos = list(  
 position = ggpp::position\_dodgenudge(x = 0, width = 0.3), width = 0.3  
),  
point.args = list(show\_guide = FALSE, alpha = .5),  
violin.args.pos = list(  
 width = 0.6, position = position\_nudge(x = 0.16)),  
point.args.pos = list(position = ggpp::position\_dodgenudge(x = -0.25, width = 0.1))) +  
 ylim(0, 700) +  
 scale\_fill\_manual(values = custom.col2) +  
 scale\_color\_manual(values = custom.col2) +  
 labs(title = "",   
 x = "",   
 y = "Reaction time (ms)") +  
 theme\_bw() +   
 theme(legend.position = "bottom",   
 plot.title = element\_text(hjust = 0.5),   
 legend.direction = "horizontal",   
 text = element\_text(size = 15))



ggsave("Fig3\_rts.svg",   
 units = "mm",   
 width = 170,  
 height = 100,  
 dpi = 300)

# S1.3 Explorative analysis of RTs considering number of saccades

Since saccadic behaviour may have influenced reaction times, we rerun the model concerning reaction times with a separate predictor coding the number of saccades of this stimulus pair.

# merge behaviour and saccades together  
df.sac.fab = merge(df.fab.full,   
 df.sac %>% group\_by(subID, diagnosis, trl) %>% summarise(n.sac = n()),   
 all.x = T) %>%  
 # compute median rt.cor  
 group\_by(subID, diagnosis, cue, stm) %>%  
 summarise(  
 rt.cor = median(rt.cor, na.rm = T),  
 n.sac = sum(n.sac, na.rm = T)  
 )

## `summarise()` has grouped output by 'subID', 'diagnosis'. You can override  
## using the `.groups` argument.  
## `summarise()` has grouped output by 'subID', 'diagnosis', 'cue'. You can  
## override using the `.groups` argument.

# set the contrasts  
contrasts(df.sac.fab$cue) = contr.sum(2)  
contrasts(df.sac.fab$cue)

## [,1]  
## face 1  
## object -1

contrasts(df.sac.fab$diagnosis) = contr.sum(4)  
contrasts(df.sac.fab$diagnosis)

## [,1] [,2] [,3]  
## ADHD 1 0 0  
## ASD 0 1 0  
## BOTH 0 0 1  
## COMP -1 -1 -1

# run the model > more iterations due to some suboptimal rhats in the first try  
set.seed(1357)  
m.rtsac = brm(rt.cor ~ diagnosis \* cue + n.sac + (cue | subID) + (cue \* diagnosis | stm),  
 df.sac.fab, prior = priors,  
 family = shifted\_lognormal,  
 iter = iter\*2, warmup = warm\*2,  
 backend = "cmdstanr", threads = threading(8), file = "m\_fab\_sac"  
 )  
rstan::check\_hmc\_diagnostics(m.rtsac$fit)

##   
## Divergences:

## 0 of 16000 iterations ended with a divergence.

##   
## Tree depth:

## 0 of 16000 iterations saturated the maximum tree depth of 10.

##   
## Energy:

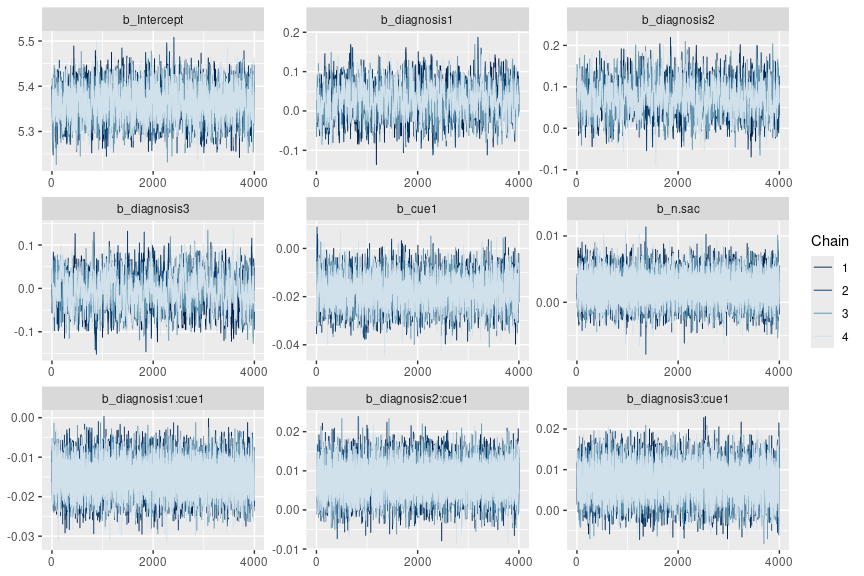
## E-BFMI indicated no pathological behavior.

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

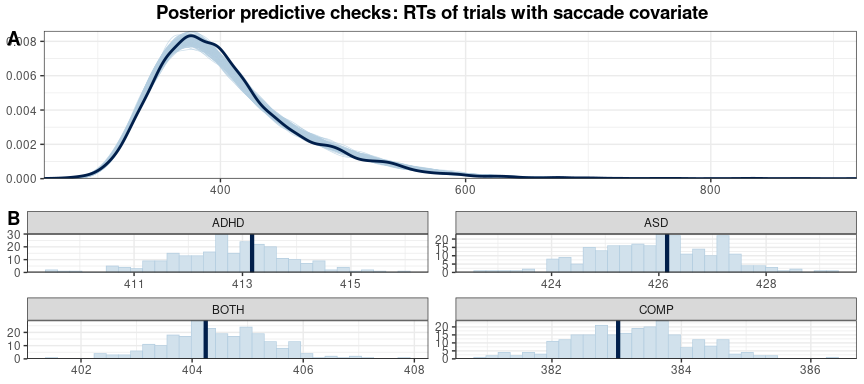
## [1] 0

# check the trace plots  
post.draws = as\_draws\_df(m.rtsac)  
mcmc\_trace(post.draws, regex\_pars = "^b\_",  
 facet\_args = list(ncol = 3)) +  
 scale\_x\_continuous(breaks=scales::pretty\_breaks(n = 3)) +  
 scale\_y\_continuous(breaks=scales::pretty\_breaks(n = 3))

## Scale for x is already present.  
## Adding another scale for x, which will replace the existing scale.



# get posterior predictions  
post.pred = posterior\_predict(m.rtsac, ndraws = nsim)  
  
# check the fit of the predicted data compared to the real data  
p1 = pp\_check(m.rtsac, ndraws = nsim) +   
 theme\_bw() + theme(legend.position = "none")  
  
df.sac.fab = df.sac.fab %>% drop\_na()  
  
# distributions of means compared to the real values per group  
p2 = ppc\_stat\_grouped(df.sac.fab$rt.cor, post.pred, df.sac.fab$diagnosis) +   
 theme\_bw() + theme(legend.position = "none")  
  
p = ggarrange(p1, p2,   
 nrow = 2, ncol = 1, labels = "AUTO")  
annotate\_figure(p,   
 top = text\_grob("Posterior predictive checks: RTs of trials with saccade covariate",   
 face = "bold", size = 14))



# print a summary  
summary(m.rtsac)

## Family: shifted\_lognormal   
## Links: mu = identity; sigma = identity; ndt = identity   
## Formula: rt.cor ~ diagnosis \* cue + n.sac + (cue | subID) + (cue \* diagnosis | stm)   
## Data: df.sac.fab (Number of observations: 6768)   
## Draws: 4 chains, each with iter = 6000; warmup = 2000; thin = 1;  
## total post-warmup draws = 16000  
##   
## Multilevel Hyperparameters:  
## ~stm (Number of levels: 36)   
## Estimate Est.Error l-95% CI u-95% CI Rhat  
## sd(Intercept) 0.03 0.00 0.02 0.04 1.00  
## sd(cue1) 0.03 0.00 0.03 0.05 1.00  
## sd(diagnosis1) 0.01 0.00 0.00 0.02 1.00  
## sd(diagnosis2) 0.00 0.00 0.00 0.01 1.00  
## sd(diagnosis3) 0.01 0.00 0.00 0.01 1.00  
## sd(cue1:diagnosis1) 0.00 0.00 0.00 0.01 1.00  
## sd(cue1:diagnosis2) 0.00 0.00 0.00 0.01 1.00  
## sd(cue1:diagnosis3) 0.01 0.00 0.00 0.01 1.00  
## cor(Intercept,cue1) -0.32 0.15 -0.60 -0.00 1.00  
## cor(Intercept,diagnosis1) -0.13 0.28 -0.65 0.45 1.00  
## cor(cue1,diagnosis1) 0.00 0.28 -0.54 0.55 1.00  
## cor(Intercept,diagnosis2) -0.09 0.31 -0.65 0.53 1.00  
## cor(cue1,diagnosis2) -0.07 0.31 -0.64 0.56 1.00  
## cor(diagnosis1,diagnosis2) -0.00 0.33 -0.62 0.63 1.00  
## cor(Intercept,diagnosis3) 0.08 0.29 -0.51 0.62 1.00  
## cor(cue1,diagnosis3) 0.21 0.29 -0.43 0.71 1.00  
## cor(diagnosis1,diagnosis3) -0.15 0.33 -0.73 0.54 1.00  
## cor(diagnosis2,diagnosis3) -0.10 0.33 -0.70 0.56 1.00  
## cor(Intercept,cue1:diagnosis1) 0.04 0.31 -0.56 0.62 1.00  
## cor(cue1,cue1:diagnosis1) -0.07 0.31 -0.63 0.54 1.00  
## cor(diagnosis1,cue1:diagnosis1) 0.00 0.33 -0.62 0.63 1.00  
## cor(diagnosis2,cue1:diagnosis1) 0.01 0.33 -0.62 0.64 1.00  
## cor(diagnosis3,cue1:diagnosis1) -0.04 0.33 -0.65 0.60 1.00  
## cor(Intercept,cue1:diagnosis2) -0.24 0.31 -0.76 0.42 1.00  
## cor(cue1,cue1:diagnosis2) -0.01 0.30 -0.58 0.56 1.00  
## cor(diagnosis1,cue1:diagnosis2) 0.04 0.33 -0.59 0.65 1.00  
## cor(diagnosis2,cue1:diagnosis2) 0.05 0.33 -0.59 0.67 1.00  
## cor(diagnosis3,cue1:diagnosis2) -0.06 0.33 -0.67 0.60 1.00  
## cor(cue1:diagnosis1,cue1:diagnosis2) -0.11 0.34 -0.72 0.58 1.00  
## cor(Intercept,cue1:diagnosis3) -0.11 0.30 -0.66 0.50 1.00  
## cor(cue1,cue1:diagnosis3) -0.18 0.30 -0.70 0.46 1.00  
## cor(diagnosis1,cue1:diagnosis3) 0.06 0.32 -0.58 0.66 1.00  
## cor(diagnosis2,cue1:diagnosis3) 0.08 0.33 -0.58 0.68 1.00  
## cor(diagnosis3,cue1:diagnosis3) -0.02 0.33 -0.65 0.61 1.00  
## cor(cue1:diagnosis1,cue1:diagnosis3) -0.09 0.34 -0.69 0.58 1.00  
## cor(cue1:diagnosis2,cue1:diagnosis3) -0.02 0.33 -0.63 0.61 1.00  
## Bulk\_ESS Tail\_ESS  
## sd(Intercept) 5427 8841  
## sd(cue1) 4975 8785  
## sd(diagnosis1) 4471 6339  
## sd(diagnosis2) 7691 8290  
## sd(diagnosis3) 5667 7105  
## sd(cue1:diagnosis1) 6625 8074  
## sd(cue1:diagnosis2) 7312 7797  
## sd(cue1:diagnosis3) 7569 8869  
## cor(Intercept,cue1) 3331 6448  
## cor(Intercept,diagnosis1) 21004 11804  
## cor(cue1,diagnosis1) 24252 11892  
## cor(Intercept,diagnosis2) 26451 11698  
## cor(cue1,diagnosis2) 28940 12033  
## cor(diagnosis1,diagnosis2) 18957 13099  
## cor(Intercept,diagnosis3) 23840 11742  
## cor(cue1,diagnosis3) 21027 10744  
## cor(diagnosis1,diagnosis3) 12065 12836  
## cor(diagnosis2,diagnosis3) 12939 13289  
## cor(Intercept,cue1:diagnosis1) 27241 12327  
## cor(cue1,cue1:diagnosis1) 27888 11914  
## cor(diagnosis1,cue1:diagnosis1) 16607 13583  
## cor(diagnosis2,cue1:diagnosis1) 13948 12579  
## cor(diagnosis3,cue1:diagnosis1) 14351 13095  
## cor(Intercept,cue1:diagnosis2) 21038 11053  
## cor(cue1,cue1:diagnosis2) 24812 11038  
## cor(diagnosis1,cue1:diagnosis2) 15771 13103  
## cor(diagnosis2,cue1:diagnosis2) 14324 13425  
## cor(diagnosis3,cue1:diagnosis2) 14336 13229  
## cor(cue1:diagnosis1,cue1:diagnosis2) 10937 13282  
## cor(Intercept,cue1:diagnosis3) 22736 11119  
## cor(cue1,cue1:diagnosis3) 24013 11918  
## cor(diagnosis1,cue1:diagnosis3) 17144 12753  
## cor(diagnosis2,cue1:diagnosis3) 14349 13395  
## cor(diagnosis3,cue1:diagnosis3) 14434 13178  
## cor(cue1:diagnosis1,cue1:diagnosis3) 11799 13519  
## cor(cue1:diagnosis2,cue1:diagnosis3) 12178 13171  
##   
## ~subID (Number of levels: 94)   
## Estimate Est.Error l-95% CI u-95% CI Rhat Bulk\_ESS Tail\_ESS  
## sd(Intercept) 0.22 0.02 0.19 0.26 1.00 2188 4453  
## sd(cue1) 0.02 0.00 0.01 0.02 1.00 7455 9127  
## cor(Intercept,cue1) -0.05 0.15 -0.34 0.24 1.00 13863 13088  
##   
## Regression Coefficients:  
## Estimate Est.Error l-95% CI u-95% CI Rhat Bulk\_ESS Tail\_ESS  
## Intercept 5.36 0.04 5.29 5.43 1.00 2277 5948  
## diagnosis1 0.03 0.04 -0.05 0.11 1.00 1144 2287  
## diagnosis2 0.07 0.04 -0.00 0.15 1.00 1262 2630  
## diagnosis3 -0.01 0.04 -0.08 0.07 1.00 1135 2355  
## cue1 -0.02 0.01 -0.03 -0.01 1.00 3645 6361  
## n.sac 0.00 0.00 -0.00 0.01 1.00 37709 11098  
## diagnosis1:cue1 -0.02 0.00 -0.02 -0.01 1.00 12937 12763  
## diagnosis2:cue1 0.01 0.00 -0.00 0.02 1.00 12764 12960  
## diagnosis3:cue1 0.01 0.00 -0.00 0.02 1.00 12667 12837  
##   
## Further Distributional Parameters:  
## Estimate Est.Error l-95% CI u-95% CI Rhat Bulk\_ESS Tail\_ESS  
## sigma 0.13 0.00 0.13 0.14 1.00 9913 10935  
## ndt 185.16 5.49 173.87 195.41 1.00 10117 10800  
##   
## 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).

# H1a: FAB effect in COMP  
h1a = hypothesis(m.rtsac,   
 "0 < 2\*(diagnosis1:cue1 + diagnosis2:cue1 + diagnosis3:cue1 - cue1)")  
h1a

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(2\*(diagnosis... < 0 -0.03 0.02 -0.06 -0.01 83.66  
## Post.Prob Star  
## 1 0.99 \*  
## ---  
## '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.

# H1b: ADHD slower than COMP  
h1b = hypothesis(m.rtsac,   
 "0 < 2\*diagnosis1 + diagnosis2 + diagnosis3")  
h1b

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(2\*diagnosis1... < 0 -0.12 0.07 -0.23 -0.02 32.61  
## Post.Prob Star  
## 1 0.97 \*  
## ---  
## '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.

# H1c: ASD slower than COMP  
h1c = hypothesis(m.rtsac,   
 "0 < 2\*diagnosis2 + diagnosis1 + diagnosis3")  
h1c

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(2\*diagnosis2... < 0 -0.17 0.06 -0.28 -0.06 227.57  
## Post.Prob Star  
## 1 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.

# H1d: FAB in ASD decreased compared to COMP  
h1d = hypothesis(m.rtsac,   
 "0 < 4\*diagnosis2:cue1 + 2\*diagnosis1:cue1 + 2\*diagnosis3:cue1")  
h1d

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(4\*diagnosis2... < 0 -0.02 0.01 -0.04 0.01 6.86  
## Post.Prob Star  
## 1 0.87   
## ---  
## '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.

# H1e: FAB in ADHD differs from FAB in COMP (undirected)  
h1e = hypothesis(m.rtsac,   
 "0 > 4\*diagnosis1:cue1 + 2\*diagnosis2:cue1 + 2\*diagnosis3:cue1",   
 alpha = 0.025)  
h1e

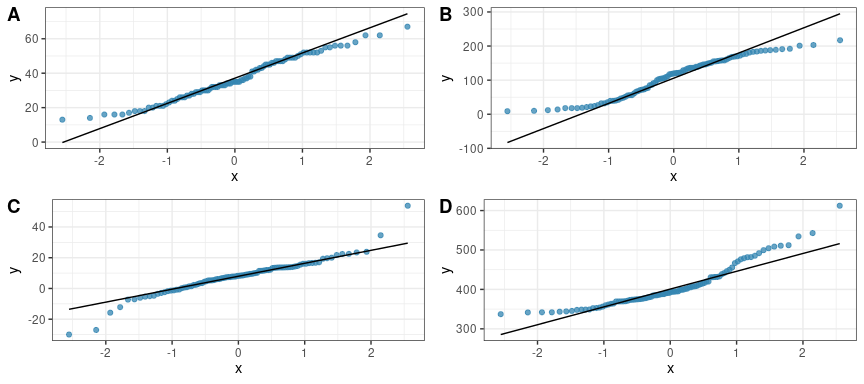
## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(4\*diagnosis1... > 0 0.03 0.01 0 0.06 73.07  
## Post.Prob Star  
## 1 0.99 \*  
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

This model confirmed the same hypotheses.

# S1.4 Explorative analysis of subject-specific FAB

## S1.4.1 Overall RT, RADS and ASRS

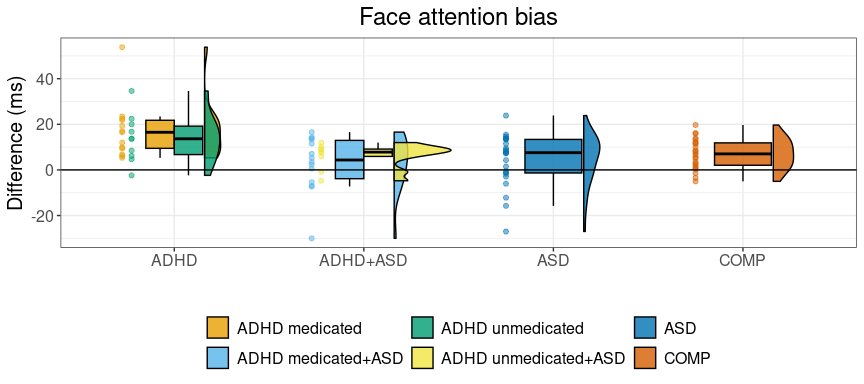
# load in gender  
df.gen = readRDS("df\_gender.rds")  
  
# merge with the questionnaire values  
df.que = df.fab.full %>%  
 group\_by(subID, diagnosis, stm, cue, ASRS\_total, RAADS\_total, adhd.meds) %>%  
 # summarise the median reaction time for each stimulus pair  
 summarise(  
 rt.cor = median(rt.cor, na.rm = T)  
 ) %>%  
 pivot\_wider(names\_from = cue, values\_from = rt.cor) %>%  
 # calculate the fab purely based on reaction times  
 mutate(  
 fab = object - face,  
 overall = (object + face) / 2   
 ) %>% group\_by(subID, diagnosis, ASRS\_total, RAADS\_total, adhd.meds) %>%  
 # calculate the mean FAB per person  
 summarise(  
 fab = mean(fab),  
 overall = mean(overall)  
 ) %>% ungroup() %>%  
 select(subID, diagnosis, overall, fab, ASRS\_total, RAADS\_total, adhd.meds) %>%  
 merge(., df.gen)  
  
# check normal distributions  
p1 = ggplot(df.que, aes(sample = ASRS\_total)) +   
 stat\_qq(alpha = 0.75, colour = c\_mid\_highlight) +   
 stat\_qq\_line() +   
 theme\_bw()  
p2 = ggplot(df.que, aes(sample = RAADS\_total)) +   
 stat\_qq(alpha = 0.75, colour = c\_mid\_highlight) +   
 stat\_qq\_line() +   
 theme\_bw()  
p3 = ggplot(df.que, aes(sample = fab)) +   
 stat\_qq(alpha = 0.75, colour = c\_mid\_highlight) +   
 stat\_qq\_line() +   
 theme\_bw()  
p4 = ggplot(df.que, aes(sample = overall)) +   
 stat\_qq(alpha = 0.75, colour = c\_mid\_highlight) +   
 stat\_qq\_line() +   
 theme\_bw()  
ggarrange(p1, p2, p3, p4,  
 nrow = 2, ncol = 2, labels = "AUTO")



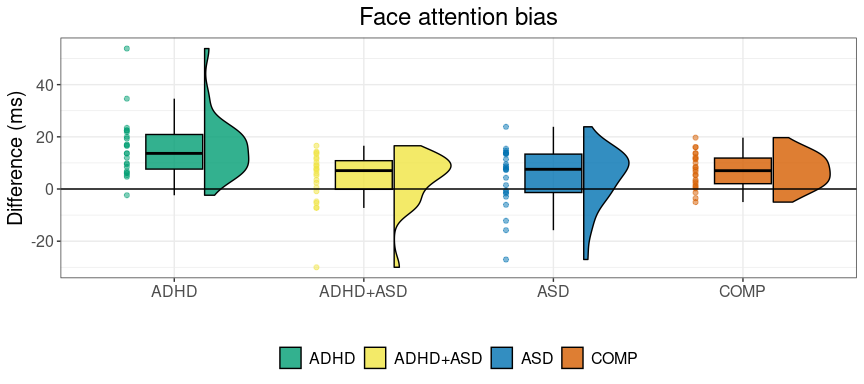
# do a Bayesian Spearman correlation: https://osf.io/j5wud  
source("./helpers/rankBasedCommonFunctions.R")  
source("./helpers/spearmanSampler.R")  
  
# Default beta prior width is set to a = b = 1 for the sampler   
if (file.exists("rho\_ASRS.rds")) {  
 rhoSamples.asrs = readRDS("rho\_ASRS.rds")  
} else {  
 set.seed(5468)  
 rhoSamples.asrs =   
 spearmanGibbsSampler(xVals = df.que$ASRS\_total,  
 yVals = df.que$fab,   
 nSamples = 5e3)  
 saveRDS(rhoSamples.asrs, file = "rho\_ASRS.rds")  
}  
if (file.exists("rho\_RADS.rds")) {  
 rhoSamples.rads = readRDS("rho\_RADS.rds")  
} else {  
 set.seed(5468)  
 rhoSamples.rads =   
 spearmanGibbsSampler(xVals = df.que$RAADS\_total,  
 yVals = df.que$fab,   
 nSamples = 5e3)  
 saveRDS(rhoSamples.rads, file = "rho\_RADS.rds")  
}  
if (file.exists("rho\_RT.rds")) {  
 rhoSamples.rt = readRDS("rho\_RT.rds")  
} else {  
 set.seed(5478)  
 rhoSamples.rt =   
 spearmanGibbsSampler(xVals = df.que$overall,  
 yVals = df.que$fab,   
 nSamples = 5e3)  
 saveRDS(rhoSamples.rt, file = "rho\_RT.rds")  
}  
  
# give the posterior samples for rho to the function below to compute BF01  
asrs.bf = computeBayesFactorOneZero(rhoSamples.asrs$rhoSamples,   
 whichTest = "Spearman",  
 priorParameter = 1)  
rads.bf = computeBayesFactorOneZero(rhoSamples.rads$rhoSamples,   
 whichTest = "Spearman",  
 priorParameter = 1)  
RT.bf = computeBayesFactorOneZero(rhoSamples.rt$rhoSamples,   
 whichTest = "Spearman",  
 priorParameter = 1)

Furthermore, Bayesian Spearman correlations revealed moderate evidence against associations between FAB and questionnaires assessing ASD (RADS: log(*BF*) = -1.63) or ADHD (ASRS: log(*BF*) = -1.61). Last, we explored whether FAB was associated with overall reaction times to assess whether attenuated FAB leads to better or worse task performance. A Bayesian Spearman correlation revealed moderate evidence against an association (log(*BF*) = -1.17).

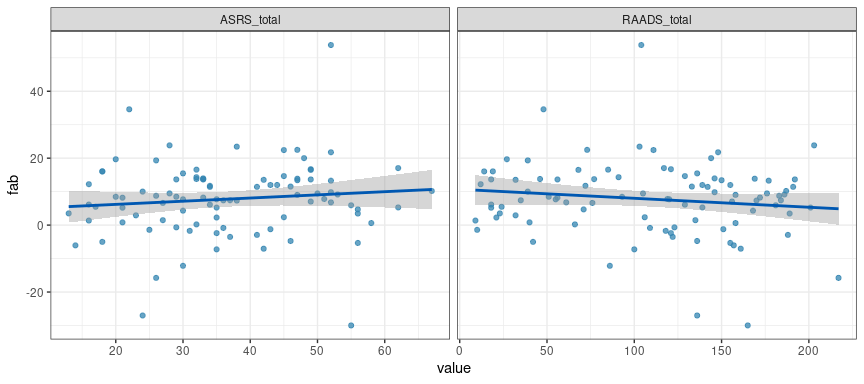
# focus on the FAB effect  
df.que %>%  
 mutate(  
 diagnosis = recode(diagnosis, "BOTH" = "ADHD+ASD"),  
 diagnosis = factor(diagnosis, levels = c("ADHD", "ADHD+ASD", "ASD", "COMP")),  
 diagnosis\_med = as.factor(case\_when(  
 adhd.meds & diagnosis == "ADHD" ~ "ADHD medicated",  
 adhd.meds & diagnosis == "ADHD+ASD" ~ "ADHD medicated+ASD",  
 diagnosis == "ADHD" ~ "ADHD unmedicated",  
 diagnosis == "ADHD+ASD" ~ "ADHD unmedicated+ASD",  
 T ~ diagnosis  
 ))  
 ) %>%   
 ggplot(aes(diagnosis, fab, fill = diagnosis\_med, colour = diagnosis\_med)) + #  
 geom\_rain(rain.side = 'r',  
boxplot.args = list(color = "black", outlier.shape = NA, show\_guide = FALSE, alpha = .8),  
violin.args = list(color = "black", outlier.shape = NA, alpha = .8),  
boxplot.args.pos = list(  
 position = ggpp::position\_dodgenudge(x = 0, width = 0.3), width = 0.3  
),  
point.args = list(show\_guide = FALSE, alpha = .5),  
violin.args.pos = list(  
 width = 0.6, position = position\_nudge(x = 0.16)),  
point.args.pos = list(position = ggpp::position\_dodgenudge(x = -0.25, width = 0.1))) +  
 geom\_hline(yintercept = 0) +  
 scale\_fill\_manual(values = custom.col) +  
 scale\_color\_manual(values = custom.col) +  
 labs(title = "Face attention bias",   
 x = "",   
 y = "Difference (ms)") +  
 theme\_bw() +   
 theme(legend.position = "bottom",   
 plot.title = element\_text(hjust = 0.5),   
 legend.direction = "horizontal",   
 text = element\_text(size = 15),  
 legend.title=element\_blank())



ggsave("Fig4\_FAB.svg",   
 units = "mm",   
 width = 170,  
 height = 100,  
 dpi = 300)  
  
# focus on the FAB effect  
df.que %>%  
 mutate(  
 diagnosis = recode(diagnosis, "BOTH" = "ADHD+ASD"),  
 diagnosis = factor(diagnosis, levels = c("ADHD", "ADHD+ASD", "ASD", "COMP"))  
 ) %>%   
 ggplot(aes(diagnosis, fab, fill = diagnosis, colour = diagnosis)) + #  
 geom\_rain(rain.side = 'r',  
boxplot.args = list(color = "black", outlier.shape = NA, show\_guide = FALSE, alpha = .8),  
violin.args = list(color = "black", outlier.shape = NA, alpha = .8),  
boxplot.args.pos = list(  
 position = ggpp::position\_dodgenudge(x = 0, width = 0.3), width = 0.3  
),  
point.args = list(show\_guide = FALSE, alpha = .5),  
violin.args.pos = list(  
 width = 0.6, position = position\_nudge(x = 0.16)),  
point.args.pos = list(position = ggpp::position\_dodgenudge(x = -0.25, width = 0.1))) +  
 geom\_hline(yintercept = 0) +  
 scale\_fill\_manual(values = custom.col[c(3, 4, 5, 6)]) +  
 scale\_color\_manual(values = custom.col[c(3, 4, 5, 6)]) +  
 labs(title = "Face attention bias",   
 x = "",   
 y = "Difference (ms)") +  
 theme\_bw() +   
 theme(legend.position = "bottom",   
 plot.title = element\_text(hjust = 0.5),   
 legend.direction = "horizontal",   
 text = element\_text(size = 15),  
 legend.title=element\_blank())



ggsave("Fig4\_FAB\_new.svg",   
 units = "mm",   
 width = 170,  
 height = 100,  
 dpi = 300)  
  
# plot the associations  
df.que %>%   
 pivot\_longer(cols = c(ASRS\_total, RAADS\_total), names\_to = "questionnaire") %>%  
 ggplot(., aes(y = fab, x = value)) +  
 geom\_point(colour = c\_mid\_highlight, alpha = 0.75) +  
 geom\_smooth(method = "lm",   
 formula = y ~ x,   
 colour = c\_dark\_highlight) +  
 facet\_grid(. ~ questionnaire, scale = "free\_x") +  
 theme\_bw()



## S1.4.2 Gender

df.que %>%   
 mutate(fab = fab > 0) %>%  
 group\_by(diagnosis, gender, fab) %>% count()

## # A tibble: 16 × 4  
## # Groups: diagnosis, gender, fab [16]  
## diagnosis gender fab n  
## <fct> <fct> <lgl> <int>  
## 1 ADHD dan TRUE 2  
## 2 ADHD fem TRUE 9  
## 3 ADHD mal FALSE 1  
## 4 ADHD mal TRUE 11  
## 5 ASD fem FALSE 4  
## 6 ASD fem TRUE 8  
## 7 ASD mal FALSE 4  
## 8 ASD mal TRUE 8  
## 9 BOTH dan TRUE 3  
## 10 BOTH fem FALSE 4  
## 11 BOTH fem TRUE 8  
## 12 BOTH mal FALSE 2  
## 13 BOTH mal TRUE 6  
## 14 COMP fem TRUE 11  
## 15 COMP mal FALSE 3  
## 16 COMP mal TRUE 10

aov.gen = anovaBF(fab ~ gender \* diagnosis, data = df.que %>% filter(gender != "dan"))  
aov.gen@bayesFactor

## bf error  
## diagnosis 3.0903949 3.147131e-05  
## gender -1.1854155 1.895700e-04  
## diagnosis + gender 1.7152052 8.341929e-03  
## diagnosis + gender + diagnosis:gender 0.2125155 5.101430e-02  
## time code  
## diagnosis Wed Jul 2 09:47:07 2025 3ffd5be01c18  
## gender Wed Jul 2 09:47:07 2025 3ffd24252f13  
## diagnosis + gender Wed Jul 2 09:47:07 2025 3ffd31aba25d  
## diagnosis + gender + diagnosis:gender Wed Jul 2 09:47:07 2025 3ffd19afced0

# S1.5 Explorative analysis of errors

Last but not least, we are going to explore possible differences with regards to mean accuracies using a bernoulli distribution.

Next, we are going to explore possible differences with regards to accuracy. We use a bernoulli distribution to model the threshold between correct and incorrect trials. We computed the SBC outside of this script in batches to avoid running out of memory. Then, we combined it and load the results in here.

## Simulation-based calibration

# figure out slopes for subject  
kable(head(df.fab.full %>% count(subID, cue)))

| subID | cue | n |
| --- | --- | --- |
| 1 | face | 216 |
| 1 | object | 216 |
| 2 | face | 216 |
| 2 | object | 216 |
| 3 | face | 216 |
| 3 | object | 216 |

kable(head(df.fab.full %>% count(stm, cue, diagnosis)))

| stm | cue | diagnosis | n |
| --- | --- | --- | --- |
| 1\_10 | face | ADHD | 138 |
| 1\_10 | face | ASD | 144 |
| 1\_10 | face | BOTH | 138 |
| 1\_10 | face | COMP | 144 |
| 1\_10 | object | ADHD | 138 |
| 1\_10 | object | ASD | 144 |

code = "FAB\_err"  
  
# increase iterations a bit to improve rhats  
iter = 4000  
warm = 2000  
  
# code accuracy to track errors  
df.fab.full = df.fab.full %>%  
 mutate(  
 error = if\_else(acc,0,1)  
 )  
  
# set the formula  
f.err = brms::bf(error ~ diagnosis \* cue + (cue | subID) + (diagnosis \* cue | stm) )  
  
# set weakly informed priors  
priors = c(  
 prior(normal(6.0, 1.00), class = Intercept),  
 prior(normal(1.0, 0.50), class = sd),  
 prior(lkj(2), class = cor),  
 # no specific expectations for the rest of the effects  
 prior(normal(0, 1.00), class = b)  
)

# check if the SBC already exists  
if (file.exists(file.path(cache\_dir, sprintf("df\_res\_%s.rds", code)))) {  
 # load in the resultsn of the SBC  
 df.results = readRDS(file.path(cache\_dir, sprintf("df\_res\_%s.rds", code)))  
 df.backend = readRDS(file.path(cache\_dir, sprintf("df\_div\_%s.rds", code)))  
 dat = readRDS(file.path(cache\_dir, sprintf("dat\_%s.rds", code)))  
} else {  
 # perform SBC  
 gen = SBC\_generator\_brms(f.err, data = df.fab.full, prior = priors,   
 thin = 50, warmup = 10000, refresh = 2000,  
 generate\_lp = TRUE, family = bernoulli, init = 0.1)  
 bck = SBC\_backend\_brms\_from\_generator(gen, chains = 4, thin = 1,  
 warmup = warm, iter = iter)  
 if (!file.exists(file.path(cache\_dir, sprintf("dat\_%s.rds", code)))) {  
 dat = generate\_datasets(gen, nsim)  
 saveRDS(dat, file.path(cache\_dir, sprintf("dat\_%s.rds", code)))  
 } else {  
 dat = readRDS(file.path(cache\_dir, sprintf("dat\_%s.rds", code)))  
 }  
 res = compute\_SBC(dat,   
 bck,  
 cache\_mode = "results",   
 cache\_location = file.path(cache\_dir, sprintf("res\_%s", code)))  
 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")))  
}

Looking at the rhats and divergent transitions shows that 0 of 250 simulations had at least one parameter that had an rhat of at least 1.05 and 2 had divergent samples. Therefore, we continue with this model and plot the simulated values to perform prior predictive checks.

# create a matrix out of generated data  
dvname = gsub(" ", "", gsub("[\\|~].\*", "", f.err)[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  
  
# compute one histogram per simulated data-set   
options = c(0, 1)  
histmat = matrix(NA, ncol = nrow(truePars), length(options))   
for (i in 1:nrow(truePars)) {  
 for (j in 1:length(options))  
 {  
 histmat[j,i] = sum(dvfakemat[,i] == options[j])  
 }  
}  
# for each bin, compute quantiles across histograms   
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)  
}  
quantmat$x = c("error", "correct")  
p0 = ggplot(data = quantmat, aes(x = x)) +   
 geom\_bar(aes(y = p0.9), fill = c\_light, stat = "identity") +   
 geom\_bar(aes(y = p0.8), fill = c\_light\_highlight, stat = "identity") +   
 geom\_bar(aes(y = p0.7), fill = c\_mid, stat = "identity") +   
 geom\_bar(aes(y = p0.6), fill = c\_mid\_highlight, stat = "identity") +   
 geom\_bar(aes(y = p0.5), fill = c\_dark, stat = "identity") +   
 labs(title = "Prior predictive distribution", y = "", x = "") +  
 theme\_bw()  
  
# get simulation numbers with issues  
check = merge(df.results %>%   
 group\_by(sim\_id) %>% summarise(rhat = max(rhat, na.rm = T), max\_rank = min(max\_rank)) %>%   
 filter(rhat >= 1.05 | max\_rank != max(max\_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))  
  
prior\_sd = setNames(rep(1, length(unique(df.results.b$variable))), # all same SD   
 unique(df.results.b$variable))  
  
p4 = plot\_contraction(df.results.b, prior\_sd = prior\_sd) + theme\_bw() +  
 scale\_x\_continuous(breaks=scales::pretty\_breaks(n = 3))  
  
p = ggarrange(p0, p1, p2, p3, p4,   
 labels = "AUTO", ncol = 1, nrow = 5,   
 heights = c(1, 2, 2, 2, 2))  
annotate\_figure(p,   
 top = text\_grob("Prior predictive checks and SBC",   
 face = "bold", size = 14))

Everything looks good with the wider priors, so we continue and run the model.

## Posterior predictive checks

As the next step, we fit the model, check whether there are divergence or rhat issues, and then check whether the chains have converged.

# fit the final model  
set.seed(1234)  
m.err = brm(f.err,  
 df.fab.full, prior = priors,  
 iter = iter, warmup = warm,  
 family = "bernoulli",  
 backend = "cmdstanr", threads = threading(8),  
 file = "m\_err",  
 save\_pars = save\_pars(all = TRUE)  
 )  
rstan::check\_hmc\_diagnostics(m.err$fit)

##   
## Divergences:

## 0 of 8000 iterations ended with a divergence.

##   
## Tree depth:

## 0 of 8000 iterations saturated the maximum tree depth of 10.

##   
## Energy:

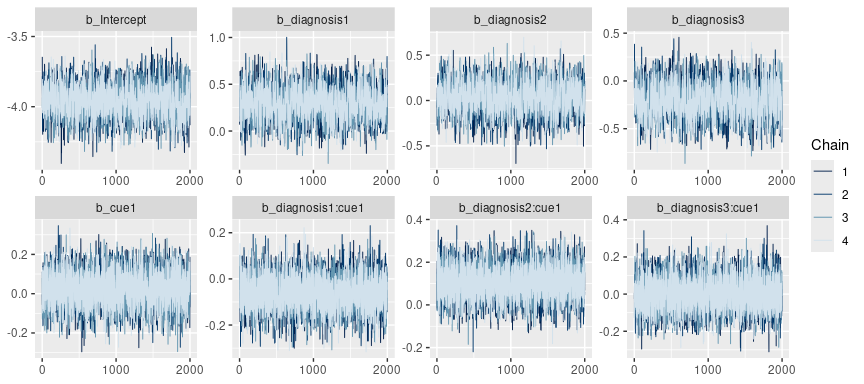
## E-BFMI indicated no pathological behavior.

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

## [1] 0

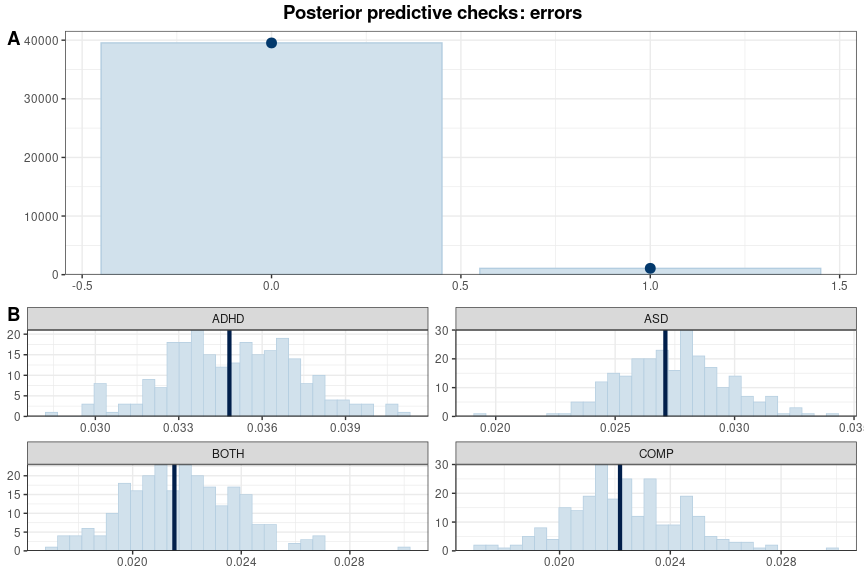
# check the trace plots  
post.draws = as\_draws\_df(m.err)  
mcmc\_trace(post.draws, regex\_pars = "^b\_",  
 facet\_args = list(ncol = 4)) +  
 scale\_x\_continuous(breaks=scales::pretty\_breaks(n = 3)) +  
 scale\_y\_continuous(breaks=scales::pretty\_breaks(n = 3))

## Scale for x is already present.  
## Adding another scale for x, which will replace the existing scale.



Again, we use the function brms::pp\_check() with 250 draws to check whether the predicted data resembles the actual data as well as the ppc\_stat\_grouped function from the bayesplot package to check posterior fit for each diagnostic group separately. The model seems to be a good fit with the predicted data closely mirroring the real data.

# get posterior predictions  
post.pred = posterior\_predict(m.err, ndraws = nsim)  
  
# check the fit of the predicted data compared to the real data  
p1 = pp\_check(m.err, ndraws = nsim, type = "bars") +   
 theme\_bw() + theme(legend.position = "none") + labs(y = "")  
  
# distributions of means compared to the real values per group  
p2 = ppc\_stat\_grouped(df.fab.full$error, post.pred, df.fab.full$diagnosis) +   
 theme\_bw() + theme(legend.position = "none")  
  
p = ggarrange(p1, p2,  
 nrow = 2, ncol = 1, labels = "AUTO")  
annotate\_figure(p,   
 top = text\_grob("Posterior predictive checks: errors",   
 face = "bold", size = 14))



Our model fits the data very well.

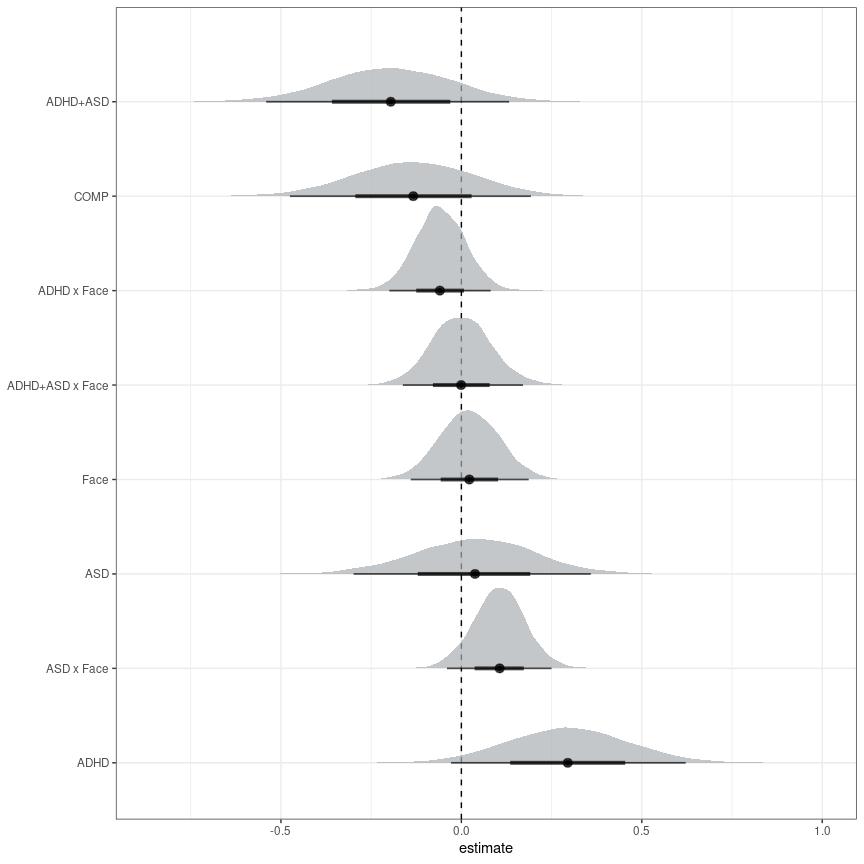
## Inferences

Now that we are convinced that we can trust our model, we have a look at its estimate and use the hypothesis function to perform explorative tests.

# print a summary  
summary(m.err)

## Family: bernoulli   
## Links: mu = logit   
## Formula: error ~ diagnosis \* cue + (cue | subID) + (diagnosis \* cue | stm)   
## Data: df.fab.full (Number of observations: 40608)   
## Draws: 4 chains, each with iter = 4000; warmup = 2000; thin = 1;  
## total post-warmup draws = 8000  
##   
## Multilevel Hyperparameters:  
## ~stm (Number of levels: 36)   
## Estimate Est.Error l-95% CI u-95% CI Rhat  
## sd(Intercept) 0.29 0.06 0.18 0.42 1.00  
## sd(diagnosis1) 0.11 0.07 0.01 0.25 1.00  
## sd(diagnosis2) 0.09 0.06 0.01 0.23 1.00  
## sd(diagnosis3) 0.10 0.07 0.00 0.26 1.00  
## sd(cue1) 0.41 0.07 0.30 0.55 1.00  
## sd(diagnosis1:cue1) 0.10 0.07 0.00 0.25 1.00  
## sd(diagnosis2:cue1) 0.08 0.06 0.00 0.22 1.00  
## sd(diagnosis3:cue1) 0.21 0.09 0.03 0.38 1.00  
## cor(Intercept,diagnosis1) -0.14 0.28 -0.64 0.43 1.00  
## cor(Intercept,diagnosis2) 0.14 0.29 -0.47 0.66 1.00  
## cor(diagnosis1,diagnosis2) -0.08 0.30 -0.64 0.51 1.00  
## cor(Intercept,diagnosis3) 0.06 0.29 -0.51 0.60 1.00  
## cor(diagnosis1,diagnosis3) -0.02 0.30 -0.60 0.56 1.00  
## cor(diagnosis2,diagnosis3) -0.07 0.31 -0.62 0.55 1.00  
## cor(Intercept,cue1) -0.14 0.20 -0.51 0.25 1.00  
## cor(diagnosis1,cue1) 0.07 0.28 -0.47 0.60 1.01  
## cor(diagnosis2,cue1) 0.14 0.29 -0.46 0.65 1.00  
## cor(diagnosis3,cue1) -0.01 0.28 -0.54 0.54 1.00  
## cor(Intercept,diagnosis1:cue1) -0.09 0.29 -0.62 0.49 1.00  
## cor(diagnosis1,diagnosis1:cue1) 0.01 0.29 -0.55 0.58 1.00  
## cor(diagnosis2,diagnosis1:cue1) -0.01 0.30 -0.59 0.57 1.00  
## cor(diagnosis3,diagnosis1:cue1) -0.02 0.30 -0.57 0.56 1.00  
## cor(cue1,diagnosis1:cue1) -0.03 0.28 -0.57 0.54 1.00  
## cor(Intercept,diagnosis2:cue1) -0.09 0.29 -0.63 0.49 1.00  
## cor(diagnosis1,diagnosis2:cue1) 0.06 0.30 -0.53 0.62 1.00  
## cor(diagnosis2,diagnosis2:cue1) -0.03 0.30 -0.58 0.56 1.00  
## cor(diagnosis3,diagnosis2:cue1) 0.04 0.30 -0.55 0.61 1.00  
## cor(cue1,diagnosis2:cue1) -0.00 0.29 -0.57 0.54 1.00  
## cor(diagnosis1:cue1,diagnosis2:cue1) -0.06 0.31 -0.63 0.54 1.00  
## cor(Intercept,diagnosis3:cue1) -0.17 0.25 -0.64 0.35 1.00  
## cor(diagnosis1,diagnosis3:cue1) -0.01 0.29 -0.57 0.55 1.00  
## cor(diagnosis2,diagnosis3:cue1) -0.00 0.30 -0.57 0.57 1.00  
## cor(diagnosis3,diagnosis3:cue1) 0.03 0.30 -0.54 0.59 1.00  
## cor(cue1,diagnosis3:cue1) 0.16 0.25 -0.34 0.62 1.00  
## cor(diagnosis1:cue1,diagnosis3:cue1) 0.02 0.29 -0.53 0.58 1.00  
## cor(diagnosis2:cue1,diagnosis3:cue1) -0.00 0.30 -0.56 0.56 1.00  
## Bulk\_ESS Tail\_ESS  
## sd(Intercept) 3331 5105  
## sd(diagnosis1) 2577 2934  
## sd(diagnosis2) 2917 3935  
## sd(diagnosis3) 2595 3064  
## sd(cue1) 4182 5911  
## sd(diagnosis1:cue1) 3358 4107  
## sd(diagnosis2:cue1) 3994 3354  
## sd(diagnosis3:cue1) 2482 2203  
## cor(Intercept,diagnosis1) 8818 5766  
## cor(Intercept,diagnosis2) 8891 5504  
## cor(diagnosis1,diagnosis2) 6537 5896  
## cor(Intercept,diagnosis3) 9817 6004  
## cor(diagnosis1,diagnosis3) 9427 6365  
## cor(diagnosis2,diagnosis3) 6602 6202  
## cor(Intercept,cue1) 3286 4640  
## cor(diagnosis1,cue1) 1136 2520  
## cor(diagnosis2,cue1) 1059 1781  
## cor(diagnosis3,cue1) 1240 2554  
## cor(Intercept,diagnosis1:cue1) 9979 5965  
## cor(diagnosis1,diagnosis1:cue1) 8566 6095  
## cor(diagnosis2,diagnosis1:cue1) 7508 5399  
## cor(diagnosis3,diagnosis1:cue1) 6392 5813  
## cor(cue1,diagnosis1:cue1) 10897 6192  
## cor(Intercept,diagnosis2:cue1) 10303 6064  
## cor(diagnosis1,diagnosis2:cue1) 8178 6098  
## cor(diagnosis2,diagnosis2:cue1) 7641 6144  
## cor(diagnosis3,diagnosis2:cue1) 7422 6547  
## cor(cue1,diagnosis2:cue1) 9720 6039  
## cor(diagnosis1:cue1,diagnosis2:cue1) 6110 6005  
## cor(Intercept,diagnosis3:cue1) 6693 5359  
## cor(diagnosis1,diagnosis3:cue1) 4803 5384  
## cor(diagnosis2,diagnosis3:cue1) 4253 5924  
## cor(diagnosis3,diagnosis3:cue1) 3807 5235  
## cor(cue1,diagnosis3:cue1) 9371 6338  
## cor(diagnosis1:cue1,diagnosis3:cue1) 4578 6707  
## cor(diagnosis2:cue1,diagnosis3:cue1) 5120 6589  
##   
## ~subID (Number of levels: 94)   
## Estimate Est.Error l-95% CI u-95% CI Rhat Bulk\_ESS Tail\_ESS  
## sd(Intercept) 0.86 0.08 0.70 1.03 1.00 2082 4075  
## sd(cue1) 0.20 0.06 0.06 0.32 1.00 1735 1440  
## cor(Intercept,cue1) 0.09 0.23 -0.37 0.53 1.00 5194 4798  
##   
## Regression Coefficients:  
## Estimate Est.Error l-95% CI u-95% CI Rhat Bulk\_ESS Tail\_ESS  
## Intercept -3.96 0.11 -4.17 -3.74 1.01 1638 3633  
## diagnosis1 0.30 0.17 -0.03 0.62 1.00 1674 3067  
## diagnosis2 0.04 0.17 -0.30 0.36 1.00 1635 2955  
## diagnosis3 -0.20 0.17 -0.54 0.13 1.00 1604 2827  
## cue1 0.02 0.08 -0.14 0.19 1.00 4409 5756  
## diagnosis1:cue1 -0.06 0.07 -0.20 0.08 1.00 5509 5863  
## diagnosis2:cue1 0.11 0.07 -0.04 0.25 1.00 5505 6178  
## diagnosis3:cue1 0.00 0.08 -0.16 0.17 1.00 5826 5175  
##   
## 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  
as\_draws\_df(m.err) %>%   
 select(starts\_with("b\_")) %>%  
 mutate(  
 b\_COMP = - b\_diagnosis1 - b\_diagnosis2 - b\_diagnosis3  
 ) %>%  
 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", "ADHD"),  
 coef = str\_replace\_all(coef, "diagnosis2", "ASD"),  
 coef = str\_replace\_all(coef, "diagnosis3", "ADHD+ASD"),  
 coef = str\_replace\_all(coef, "cue1", "Face"),  
 coef = fct\_reorder(coef, desc(estimate))  
 ) %>%   
 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(c\_light, c\_dark)) + theme(legend.position = "none")



# COMP < ADHD  
e1 = hypothesis(m.err, "0 < 2\*diagnosis1 + diagnosis2 + diagnosis3", alpha = 0.025)  
e1

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(2\*diagnosis1... < 0 -0.43 0.27 -0.98 0.1 16.78  
## Post.Prob Star  
## 1 0.94   
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

# COMP < ASD  
e2 = hypothesis(m.err, "0 < 2\*diagnosis2 + diagnosis1 + diagnosis3", alpha = 0.025)  
e2

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(2\*diagnosis2... < 0 -0.17 0.28 -0.72 0.37 2.71  
## Post.Prob Star  
## 1 0.73   
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

# COMP < BOTH  
e3 = hypothesis(m.err, "0 > 2\*diagnosis3 + diagnosis1 + diagnosis2", alpha = 0.025)  
e3

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio  
## 1 (0)-(2\*diagnosis3... > 0 0.06 0.28 -0.49 0.62 1.43  
## Post.Prob Star  
## 1 0.59   
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

# explore differences between cues  
e4 = hypothesis(m.err, "0 > 2\*cue1", alpha = 0.025)  
e4

## Hypothesis Tests for class b:  
## Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio Post.Prob  
## 1 (0)-(2\*cue1) > 0 -0.04 0.17 -0.37 0.28 0.66 0.4  
## Star  
## 1   
## ---  
## 'CI': 95%-CI for one-sided and 97.5%-CI for two-sided hypotheses.  
## '\*': For one-sided hypotheses, the posterior probability exceeds 97.5%;  
## for two-sided hypotheses, the value tested against lies outside the 97.5%-CI.  
## Posterior probabilities of point hypotheses assume equal prior probabilities.

# extract predicted differences  
df.new = df.fab.full %>%   
 select(diagnosis, cue) %>%   
 mutate(  
 condition = paste0(diagnosis, '\_', cue)  
 ) %>%  
 distinct()  
df.ms = as.data.frame(  
 fitted(m.err, summary = F,   
 newdata = df.new,   
 re\_formula = NA))  
colnames(df.ms) = df.new$condition  
  
st(df.ms,   
 summ = c('mean(x)','sd(x)','min(x)','pctile(x)[2.5]',  
 'pctile(x)[97.5]','max(x)'))

Summary Statistics

| Variable | Mean | Sd | Min | Pctile[2.5] | Pctile[97.5] | Max |
| --- | --- | --- | --- | --- | --- | --- |
| ADHD\_face | 0.025 | 0.0054 | 0.0097 | 0.015 | 0.036 | 0.061 |
| ADHD\_object | 0.027 | 0.0058 | 0.012 | 0.017 | 0.039 | 0.059 |
| COMP\_face | 0.016 | 0.0039 | 0.0068 | 0.0098 | 0.025 | 0.045 |
| COMP\_object | 0.017 | 0.004 | 0.007 | 0.01 | 0.026 | 0.039 |
| ASD\_face | 0.023 | 0.005 | 0.0093 | 0.014 | 0.033 | 0.046 |
| ASD\_object | 0.018 | 0.004 | 0.0069 | 0.011 | 0.026 | 0.042 |
| BOTH\_face | 0.016 | 0.0039 | 0.0065 | 0.0094 | 0.025 | 0.041 |
| BOTH\_object | 0.016 | 0.0037 | 0.0054 | 0.0092 | 0.023 | 0.033 |

st(df.ms %>%   
 mutate(  
 face = rowMeans(select(., matches(".\*\_face")), na.rm = T),  
 object = rowMeans(select(., matches(".\*\_object")), na.rm = T),  
 FAB = object - face  
 ) %>% select(face, object, FAB),  
 summ = c('mean(x)','sd(x)','min(x)','pctile(x)[2.5]','pctile(x)[97.5]','max(x)'))

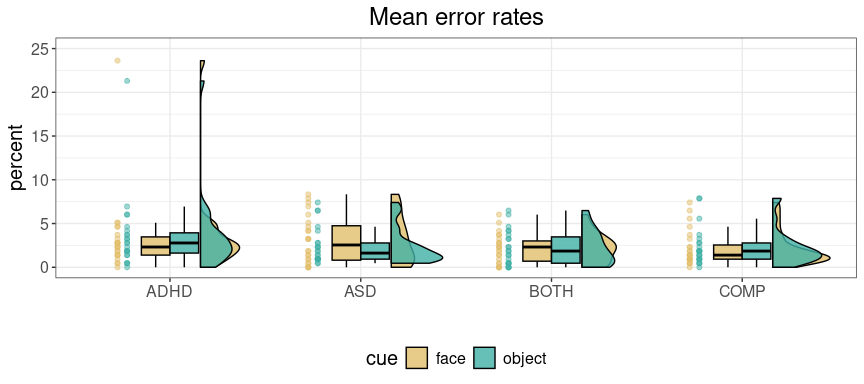
Summary Statistics

| Variable | Mean | Sd | Min | Pctile[2.5] | Pctile[97.5] | Max |
| --- | --- | --- | --- | --- | --- | --- |
| face | 0.02 | 0.0027 | 0.012 | 0.015 | 0.025 | 0.031 |
| object | 0.019 | 0.0027 | 0.012 | 0.014 | 0.025 | 0.03 |
| FAB | -0.00077 | 0.0032 | -0.013 | -0.0075 | 0.0052 | 0.012 |

Accuracies were generally high, with a total of 2.64% inaccurate responses across diagnostic groups. The explorative analysis of the error rates revealed no credible differences between any of the diagnostic groups and no credible difference between cues (see supplementary materials).

## Plots

# overall accuracies  
df.fab.full %>%   
 group\_by(subID, diagnosis, cue) %>%   
 summarise(error = 100\*mean(error, na.rm = T)) %>%   
 ggplot(aes(diagnosis, error, fill = cue, colour = cue)) + #  
 geom\_rain(rain.side = 'r',  
boxplot.args = list(color = "black", outlier.shape = NA, show\_guide = FALSE, alpha = .8),  
violin.args = list(color = "black", outlier.shape = NA, alpha = .8),  
boxplot.args.pos = list(  
 position = ggpp::position\_dodgenudge(x = 0, width = 0.3), width = 0.3  
),  
point.args = list(show\_guide = FALSE, alpha = .5),  
violin.args.pos = list(  
 width = 0.6, position = position\_nudge(x = 0.16)),  
point.args.pos = list(position = ggpp::position\_dodgenudge(x = -0.25, width = 0.1))) +  
 ylim(0, 25) +  
 scale\_fill\_manual(values = custom.col2) +  
 scale\_color\_manual(values = custom.col2) +  
 labs(title = "Mean error rates", x = "", y = "percent") +  
 theme\_bw() +   
 theme(legend.position = "bottom",   
 plot.title = element\_text(hjust = 0.5),   
 legend.direction = "horizontal",   
 text = element\_text(size = 15))



# S1.5 Summary table

# get grand average of accuracies and reaction times  
df.agg = rbind(  
 df.fab.full %>%  
 group\_by(subID, diagnosis, cue) %>%   
 summarise(error = 100\*mean(error, na.rm = T)) %>%   
 group\_by(diagnosis, cue) %>%   
 summarise(mean = mean(error, na.rm = T), se = sd(error, na.rm = T)/sqrt(n())) %>%  
 mutate(measure = "accuracy") %>%  
 mutate(  
 value = sprintf("%.2f ±%.2f", mean, se)  
 ) %>% select(measure, diagnosis, cue, value) %>%  
 pivot\_wider(names\_from = c(diagnosis, cue), values\_from = value),  
 df.fab.full %>%  
 group\_by(subID, diagnosis, cue) %>%   
 summarise(rt.cor = mean(rt.cor, na.rm = T)) %>%   
 group\_by(diagnosis, cue) %>%   
 summarise(mean = mean(rt.cor, na.rm = T), se = sd(rt.cor, na.rm = T)/sqrt(n())) %>%   
 mutate(measure = "rt.cor") %>%  
 mutate(  
 value = sprintf("%.0f ±%.0f", mean, se)  
 ) %>% select(measure, diagnosis, cue, value) %>%  
 pivot\_wider(names\_from = c(diagnosis, cue), values\_from = value))  
  
read\_docx() %>%  
 body\_add\_table(df.agg) %>%  
 print(target = "FAB\_tbl2.docx")