```
knitr::opts_chunk$set(message = FALSE, warning = FALSE)
```

```
############### 0 - safe choice A, 1 - risky choice B ####################
library(rstan); rstan_options(javascript=FALSE)
options(mc.cores = parallel::detectCores())
rstan_options(auto_write = T)
library(dplyr)



dat <- read.csv('final_data.csv')

dat <- dat %>%
  filter(skew != 'control')
dat <- dat %>%
  mutate(cho = ifelse(true_response == 'f', 1, -1))


ids <- unique(dat$Prolific_ID)
for(j in 1:length(ids)){
  dat$tid[dat$Prolific_ID==ids[j]] <- j
}
tids <- unique(dat$tid)

dat <- dat %>%
  filter(test_part == 'cs' | test_part == 'sc')

dat <- dat %>%
  mutate(
    oa_complex = ifelse(test_part == 'cs', 1, -1),
    evd = evd * oa_complex,
    sdd = sdd * oa_complex,
    chose_complex = ifelse((oa_complex == 1 & cho == 1) | (oa_complex == -1 & cho
== -1), 1, -1)
  )
```

```
dat$trialtype1 <- ifelse(dat$skew == "ns", -1, ifelse(dat$skew == "lr", 1, ifelse(
dat$skew == "rl", 0, NA)))
dat$trialtype2 <- ifelse(dat$skew == "ns", -1, ifelse(dat$skew == "lr", 0, ifelse(
dat$skew == "rl", 1, NA)))


dat$rt <- dat$rt/1000


# Assuming your dataframe is named 'df'
dat$P_A1 <- dat$P_A1 / 100
dat$P_A2 <- dat$P_A2 / 100
dat$P_B1 <- dat$P_B1 / 100
dat$P_B2 <- dat$P_B2 / 100
```

```
dataList  = list(cho = dat$cho, rt = dat$rt, participant = dat$tid,N=nrow(dat),  L
= length(tids),starting_point=0.5, evd = dat$evd, sdd = dat$sdd, trialtype1 = dat$
trialtype1, trialtype2 = dat$trialtype2)
```

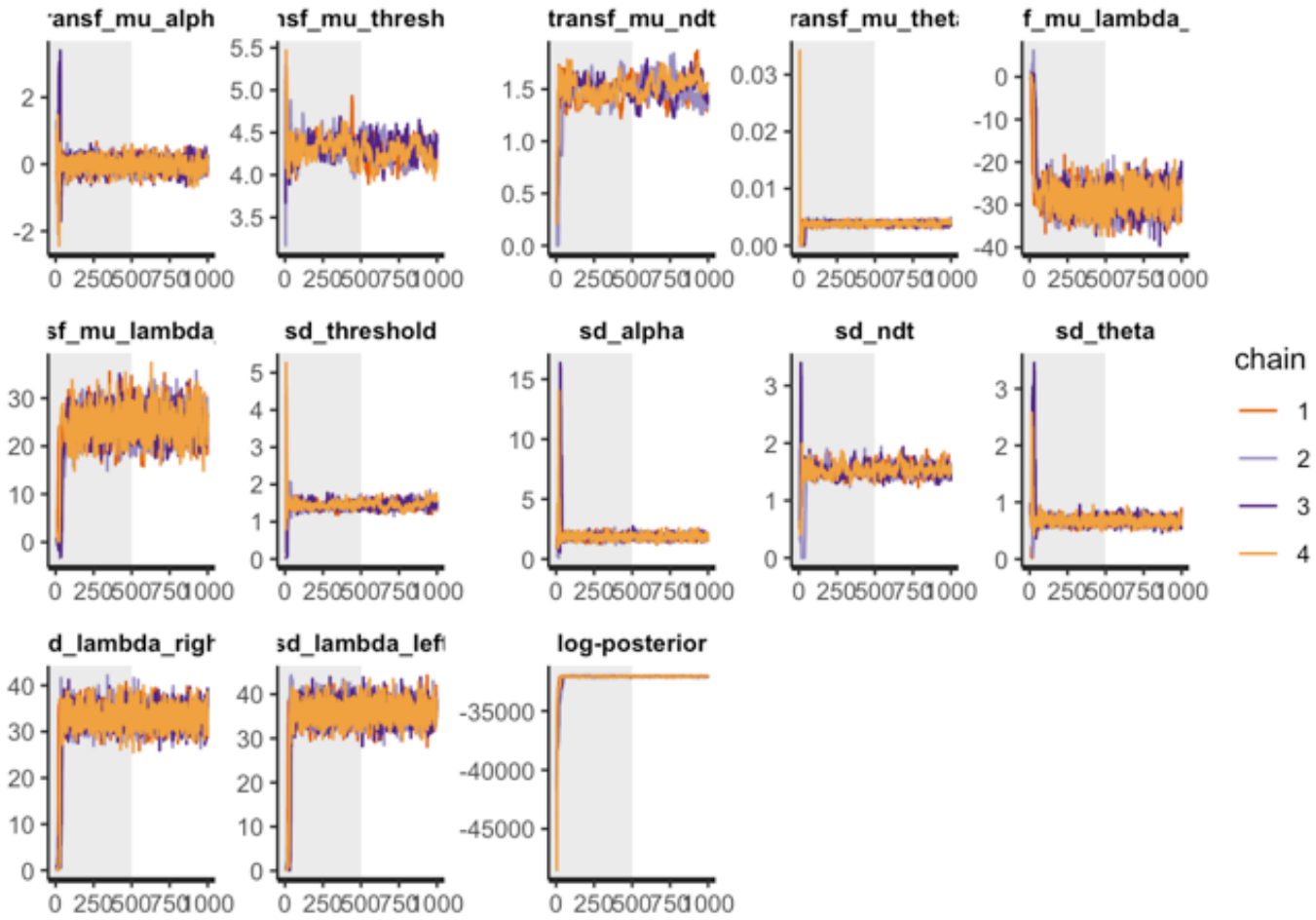
```
parameters = c("transf_mu_alpha","transf_mu_threshold","transf_mu_ndt", "transf_mu
_theta",'transf_mu_lambda_right','transf_mu_lambda_left', 'sd_threshold',"sd_alpha
","sd_ndt", 'sd_theta', 'sd_lambda_right', 'sd_lambda_left', "alpha_sbj","threshol
d_sbj","ndt_sbj",'theta_sbj', 'lambda_right_sbj','lambda_left_sbj', "log_lik")

initFunc <-function (i) {
  initList=list()
  for (ll in 1:i){
    initList[[ll]] = list(
                          mu_alpha = runif(1,-5,5),
                          sd_alpha = runif(1,0,1),
                          mu_threshold = runif(1,-0.5,5),
                          sd_threshold = runif(1,0,1),
                          mu_ndt = runif(1, -1.5, 0),
                          sd_ndt = runif(1, 0, 1),
                          mu_theta = runif(1,-20, 1),
                          sd_theta = runif(1,0,1),
                          mu_lambda_right = runif(1,-1, 1),
                          sd_lambda_right = runif(1, 0, 1),
                          mu_lambda_left = runif(1,-1, 1),
                          sd_lambda_left = runif(1, 0, 1),
                          z_alpha = runif(length(tids),-0.1,0.1),
                          z_theta = runif(length(tids),-0.1,0.1),
                          z_threshold = runif(length(tids),-0.1,0.1),
                          z_ndt = runif(length(tids),-0.1,0.1),
                          z_lambda_right = runif(length(tids),-0.1,0.1),
                          z_lambda_left = runif(length(tids),-0.1,0.1)


    )
  }

  return(initList)
}
```

```
m <- stan_model("MV_prob cs.stan")
```

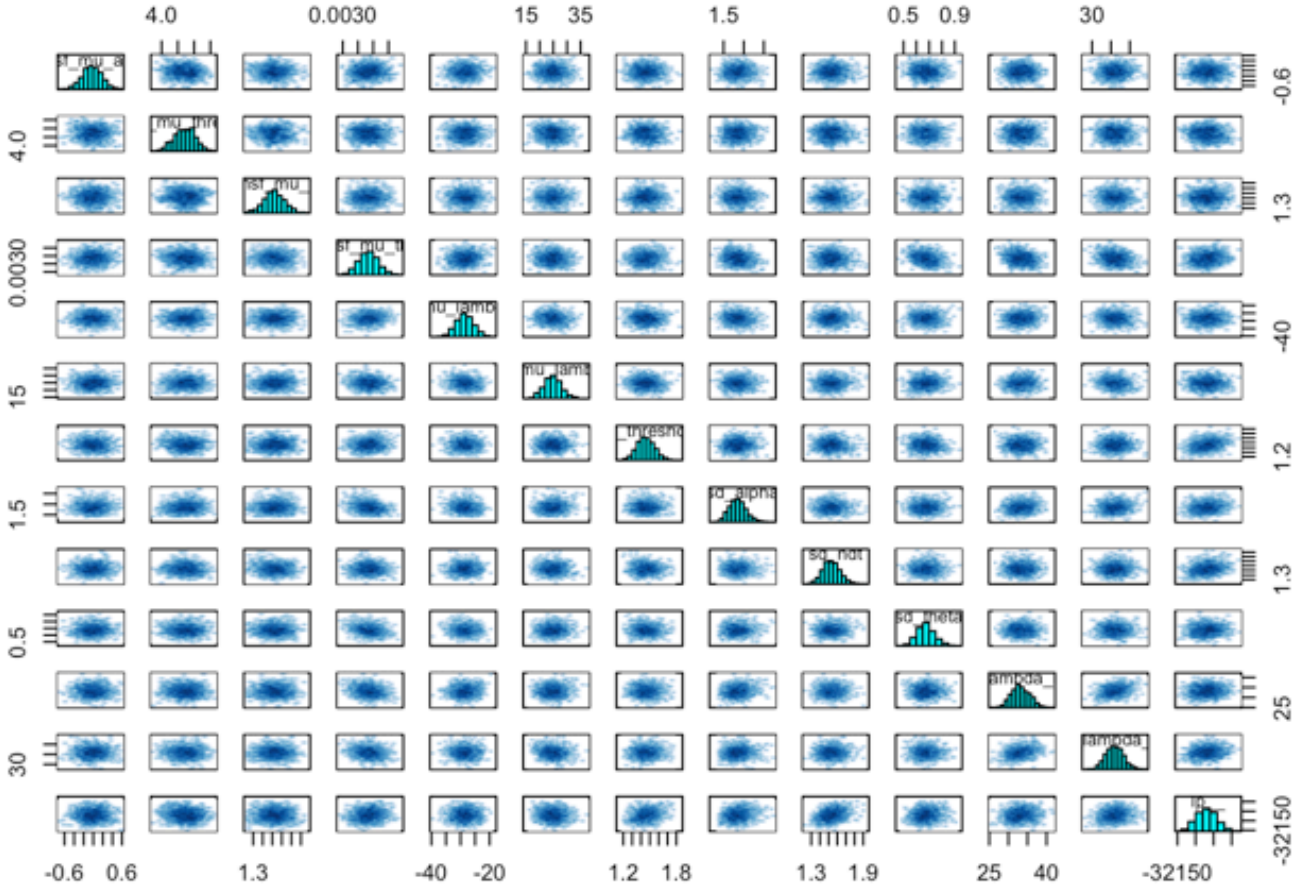
```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## clang -mmacosx-version-min=10.13 -I"/Library/Frameworks/R.framework/Resources/i
nclude" -DNDEBUG   -I"/Library/Frameworks/R.framework/Versions/4.2/Resources/libra
ry/Rcpp/include/"  -I"/Library/Frameworks/R.framework/Versions/4.2/Resources/libra
ry/RcppEigen/include/"  -I"/Library/Frameworks/R.framework/Versions/4.2/Resources/
library/RcppEigen/include/unsupported"  -I"/Library/Frameworks/R.framework/Version
s/4.2/Resources/library/BH/include" -I"/Library/Frameworks/R.framework/Versions/4.
2/Resources/library/StanHeaders/include/src/"  -I"/Library/Frameworks/R.framework/
Versions/4.2/Resources/library/StanHeaders/include/"  -I"/Library/Frameworks/R.fra
mework/Versions/4.2/Resources/library/RcppParallel/include/"  -I"/Library/Framewor
ks/R.framework/Versions/4.2/Resources/library/rstan/include" -DEIGEN_NO_DEBUG  -DB
OOST_DISABLE_ASSERTS  -DBOOST_PENDING_INTEGER_LOG2_HPP  -DSTAN_THREADS  -DUSE_STAN
C3 -DSTRICT_R_HEADERS  -DBOOST_PHOENIX_NO_VARIADIC_EXPRESSION  -D_HAS_AUTO_PTR_ETC
=0  -include '/Library/Frameworks/R.framework/Versions/4.2/Resources/library/StanH
eaders/include/stan/math/prim/fun/Eigen.hpp'  -D_REENTRANT -DRCPP_PARALLEL_USE_TBB
=1   -I/usr/local/include   -fPIC  -Wall -g -O2  -c foo.c -o foo.o
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.2/Resources/li
brary/StanHeaders/include/stan/math/prim/fun/Eigen.hpp:22:
## In file included from /Library/Frameworks/R.framework/Versions/4.2/Resources/li
brary/RcppEigen/include/Eigen/Dense:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.2/Resources/li
brary/RcppEigen/include/Eigen/Core:88:
## /Library/Frameworks/R.framework/Versions/4.2/Resources/library/RcppEigen/includ
e/Eigen/src/Core/util/Macros.h:628:1: error: unknown type name 'namespace'
## namespace Eigen {
## ^
## /Library/Frameworks/R.framework/Versions/4.2/Resources/library/RcppEigen/includ
e/Eigen/src/Core/util/Macros.h:628:16: error: expected ';' after top level declara
tor
## namespace Eigen {
##                ^
##                ;
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.2/Resources/li
brary/StanHeaders/include/stan/math/prim/fun/Eigen.hpp:22:
## In file included from /Library/Frameworks/R.framework/Versions/4.2/Resources/li
brary/RcppEigen/include/Eigen/Dense:1:
## /Library/Frameworks/R.framework/Versions/4.2/Resources/library/RcppEigen/includ
e/Eigen/Core:96:10: fatal error: 'complex' file not found
## #include <complex>
##          ^~~~~~~~
## 3 errors generated.
## make: *** [foo.o] Error 1
```

```
dsamples <- sampling(m,
                data=dataList,
                pars=parameters,
                iter=1000,
                chains=4,#If not specified, gives random inits
                init = initFunc(4),
                warmup = 500,  # Stands for burn-in; Default = iter/2
                seed = 12,  # Setting seed; Default is random seed
                refresh = 0
                )
```

```
#parameters = c("transf_mu_alpha","transf_mu_threshold","transf_mu_ndt", "transf_mu_theta",'sd_threshold',"sd_alpha","sd_ndt", 'sd_theta', "alpha_sbj","threshold_sbj","ndt_sbj",'theta_sbj',"log_lik")

rstan::traceplot(dsamples, pars=c("transf_mu_alpha","transf_mu_threshold","transf_mu_ndt", "transf_mu_theta",'transf_mu_lambda_right','transf_mu_lambda_left', 'sd_threshold',"sd_alpha","sd_ndt", 'sd_theta', 'sd_lambda_right','sd_lambda_left', "lp__"), inc_warmup = TRUE, nrow = 3)
```



```
pairs(dsamples, pars = c( "transf_mu_alpha","transf_mu_threshold","transf_mu_ndt",
"transf_mu_theta",'transf_mu_lambda_right','transf_mu_lambda_left', 'sd_threshol
d',"sd_alpha","sd_ndt", 'sd_theta', 'sd_lambda_right','sd_lambda_left', "lp__"))
```

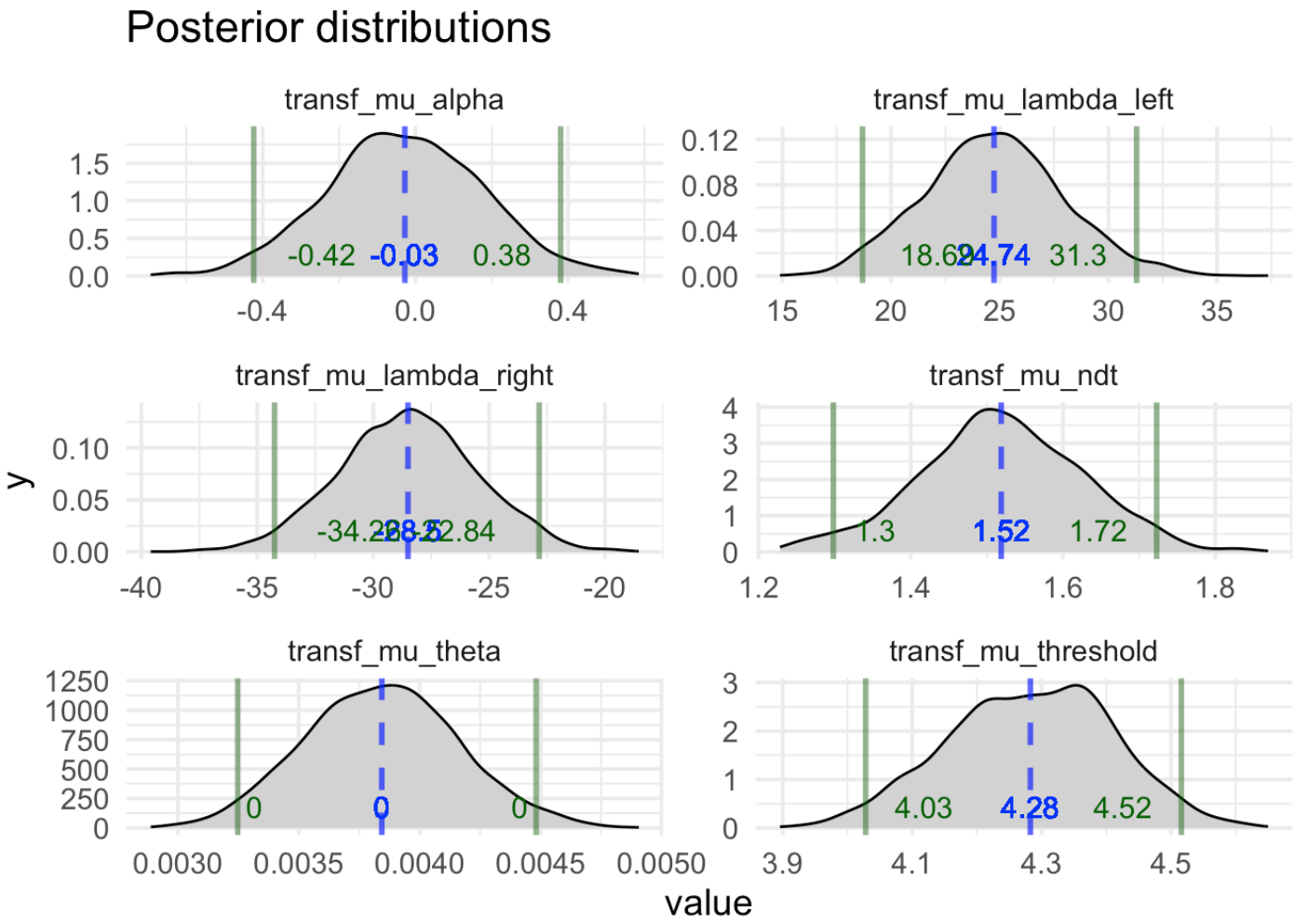



```
print(dsamples, pars = c("transf_mu_alpha","transf_mu_threshold","transf_mu_ndt",
"transf_mu_theta",'transf_mu_lambda_right','transf_mu_lambda_left', 'sd_threshol
d',"sd_alpha","sd_ndt", 'sd_theta', 'sd_lambda_right','sd_lambda_left',"lp__"))
```

```
## Inference for Stan model: anon_model.
## 4 chains, each with iter=1000; warmup=500; thin=1;
## post-warmup draws per chain=500, total post-warmup draws=2000.
##
##                            mean se_mean    sd      2.5%       25%       50%
## transf_mu_alpha           -0.03    0.01  0.20     -0.42     -0.16     -0.03
## transf_mu_threshold        4.28    0.01  0.13      4.03      4.20      4.29
## transf_mu_ndt              1.52    0.01  0.11      1.30      1.45      1.52
## transf_mu_theta            0.00    0.00  0.00      0.00      0.00      0.00
## transf_mu_lambda_right   -28.50    0.09  2.97    -34.26    -30.45    -28.52
## transf_mu_lambda_left     24.74    0.09  3.20     18.69     22.60     24.71
## sd_threshold               1.46    0.01  0.10      1.27      1.39      1.45
## sd_alpha                   1.85    0.01  0.22      1.47      1.70      1.84
## sd_ndt                     1.53    0.01  0.10      1.34      1.46      1.53
## sd_theta                   0.67    0.00  0.07      0.55      0.63      0.67
## sd_lambda_right           33.20    0.06  2.39     28.86     31.50     33.07
## sd_lambda_left            35.96    0.06  2.37     31.57     34.32     35.93
## lp__                  -32069.67    1.36 27.25 -32120.59 -32089.34 -32070.01
##                             75%     97.5% n_eff Rhat
## transf_mu_alpha            0.11      0.38  1003 1.00
## transf_mu_threshold        4.37      4.52    72 1.06
## transf_mu_ndt              1.59      1.72    66 1.03
## transf_mu_theta            0.00      0.00   627 1.00
## transf_mu_lambda_right   -26.61    -22.84  1122 1.01
## transf_mu_lambda_left     26.79     31.30  1239 1.00
## sd_threshold               1.52      1.66   111 1.02
## sd_alpha                   1.99      2.32   873 1.00
## sd_ndt                     1.60      1.74   215 1.01
## sd_theta                   0.72      0.82   616 1.01
## sd_lambda_right           34.91     37.95  1709 1.00
## sd_lambda_left            37.54     40.75  1528 1.00
## lp__                  -32051.00 -32013.66   399 1.00
##
## Samples were drawn using NUTS(diag_e) at Sun Jan 14 22:19:38 2024.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```
library(ggplot2)
library(tidyverse) # for the gather function

samples_matrix <- as.matrix(dsamples)
means <- colMeans(samples_matrix)
hpd_interval <- t(apply(samples_matrix, 2, function(x) quantile(x, probs=c(0.025,
0.975))))



parameters <- c("transf_mu_alpha","transf_mu_threshold","transf_mu_ndt", "transf_m
```

```
u_theta",'transf_mu_lambda_right', 'transf_mu_lambda_left')

# Reshape data to a long format
df_long <- as.data.frame(samples_matrix) %>%
  gather(key = "parameter", value = "value", parameters)

# Convert hpd_interval to a data frame and name the columns
hpd_interval_sub <- hpd_interval[parameters, ]
hpd_df <- as.data.frame(hpd_interval_sub)
colnames(hpd_df) <- c("lower", "upper")
rownames(hpd_df) <- parameters
hpd_df$parameter <- rownames(hpd_df)



# Aesthetic enhancements
theme_set(theme_minimal(base_size = 14)) # Set the default theme

custom_palette <- c("density_fill" = "lightgray",
                    "mean_line" = "blue",
                    "hpd_line" = "darkgreen")

# Add text labels for mean, lower, and upper HPD values
df_long <- df_long %>%
  group_by(parameter) %>%
  mutate(mean = means[parameter])

hpd_df <- hpd_df %>%
  mutate(mid = (lower + upper) / 2)

p <- ggplot(df_long, aes(x = value)) +
  geom_density(aes(fill = "density_fill")) +
  scale_fill_manual(values = custom_palette, guide = FALSE) +
  geom_vline(aes(xintercept = mean, color = "mean_line"), linetype = "dashed", siz
e = 1, alpha = 0.7) +
  geom_text(data = df_long, aes(x = mean, y = 0, label = round(mean, 2)), vjust =
-0.5, hjust = 0.5, size = 4, color = custom_palette["mean_line"]) +
  geom_vline(data = hpd_df, aes(xintercept = lower, color = "hpd_line"), linetype
= "solid", size = 1, alpha = 0.5) +
  geom_text(data = hpd_df, aes(x = lower, y = 0, label = round(lower, 2)), vjust =
-0.5, hjust = -0.5, size = 4, color = custom_palette["hpd_line"]) +
  geom_vline(data = hpd_df, aes(xintercept = upper, color = "hpd_line"), linetype
= "solid", size = 1, alpha = 0.5) +
  geom_text(data = hpd_df, aes(x = upper, y = 0, label = round(upper, 2)), vjust =
-0.5, hjust = 1.5, size = 4, color = custom_palette["hpd_line"]) +
  facet_wrap(~ parameter, scales = "free", ncol = 2) +
  scale_color_manual(values = custom_palette, guide = 'none') +
  labs(title = "Posterior distributions")

print(p)
```


Posterior distributions
transf_mu_alpha
-0.42
-0.03
0.38
transf_mu_lambda_left
18.62
24.74
31.3
transf_mu_lambda_right
-34.28
-28.5
-22.84
transf_mu_ndt
1.3
1.52
1.72
transf_mu_theta
0
0
0
transf_mu_threshold
4.03
4.28
4.52
y
value