Diminishing Self-Prioritization: Enhancing Social Perception through Reverse Matching Training

Xunqing Zheng & Chenghao Zhou

Introduction

In social interactions, individuals often navigate a complex landscape of interpersonal dynamics. Human cognition is inherently biased towards prioritizing one's own interests and viewpoints, potentially hindering the capacity for empathetic and socially attuned behavior. This phenomenon, known as self-prioritization, which has significant implications for the quality of interpersonal relationships, cooperation, and societal harmony. Reverse Matching Training targets the underlying biases associated with self-prioritization directly and aims to reorient individuals' cognitive processes towards a more equitable consideration of others' perspectives.

In this longitudinal cognitive intervention experiment, fourteen participants were divided into two groups based on their task prioritization: match-priority and mismatch-priority. Initially, they learned to link specific shapes - circles and squares - with textual labels, including "self," "other," and two visually similar Chinese characters. During the testing phase, participants responded to a visual cue - a fixation cross for 500-800 ms - followed by simultaneous presentation of a shape and text. They had to quickly decide if the combination was a match or mismatch based on their initial training, using specific keys for each condition. The match-priority group was instructed to press one key if the items matched and another if they did not match or if it was a filler item. Conversely, the mismatch-priority group was to press one key for mismatches and another for matches or fillers. This was followed by a 1200 ms blank screen and a 1500 ms relaxation period. The display used was a 1024x768 CRT monitor refreshing at 85Hz. Assessments were made at baseline, across seven training sessions, and during a formal experimental phase.

Initial Project Proposal

This project analysis is aiming to understand the data from a Bayesian approach to answer following questions.

1. Does task prioritization affect reaction and accuracy in cognitive matching tasks?
2. Are there observable changes in EEG patterns related to task accuracy?
3. Do participants improve in task performance over repeated sessions?

Initially, we were thinking about analyzing the data with Bayesian binomial model to estimate the accuracy/reaction time for each group, than apply Bayesian multivariate analysis to correlate EEG data (frequency bands, amplitude, etc.) with task performance (correct/incorrect).

However, after lengthy discussion and examining the available raw data, we decided to focus this analysis project on participant's reaction time and to understand its improvement after training. What's worth calling out is that for the simplicity of this project, we did not include the accuracy in this analysis

Packages

```r
library(readr)
library(dplyr)
library(tidyr)
library(ggplot2)
library(brms)
library(emmeans)
library(loo)
library(rstanarm)
```

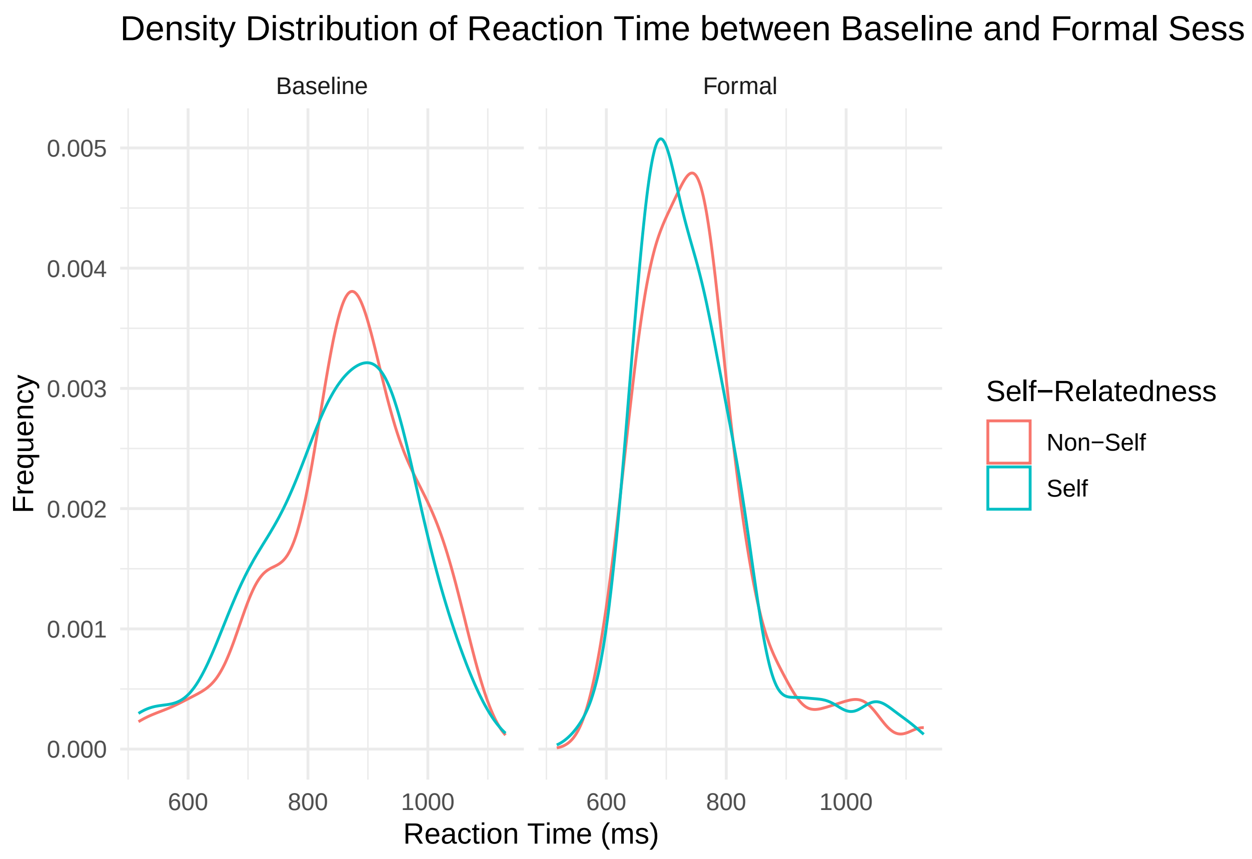
Raw Data Summary

The first step into the analysis is data cleaning, which we first converted the average reaction time from seconds to milliseconds as most of this types of analysis were done in milliseconds. In addition to that, we also make sure all the necessary conditions are clearly identified in the data, such as if the specific row is associated with "Fill", "Match", or "Mismatch" experiment session, and if the session is self related or not. An intersection condition column is created based on these conditions: 1 = Fill Self, 2 = Mismatch Self, 3 = Match Self, 4 = Fill Non-Self, 5 = Mismatch Non-Self, 6 = Match Non-Self. Lastly, we filtered data to only focused on only the reaction time at Baseline and the Formal tests, removing all the training sections in between.

```r
#clean data
RawRT = read_csv("RawRT.csv")
#data <- read_csv("UsefulData05122024.csv")

#filtered_data <- data %>%
filtered_data <- RawRT %>%
  mutate(
    meanRT = meanRT * 1000,  # Convert to milliseconds
    session = factor(session, levels = c(1:9), labels = c("Baseline", 'TR1',
                                                          'TR2', 'TR3', 'TR4',
                                                          'TR5', 'TR6', 'TR7',
                                                          "Formal")),
    group = factor(group, levels = c(1, 2), labels = c("Match First",
                                                       "Mismatch First")),
    type = case_when(
      conds %in% c(1, 4) ~ "Fill",
      conds %in% c(3, 6) ~ "Match",
      conds %in% c(2, 5) ~ "Mismatch"
    ),
    ifself_related = case_when(
      conds %in% c(1, 2, 3) ~ "Self",
      conds %in% c(4, 5, 6) ~ "Non-Self"
    ),
    type = factor(type),
    ifself_related = factor(ifself_related),
    gender = factor(gender)  # Ensure gender is treated as a categorical variable
  ) %>%
  filter(session %in% c("Baseline", "Formal"))
```

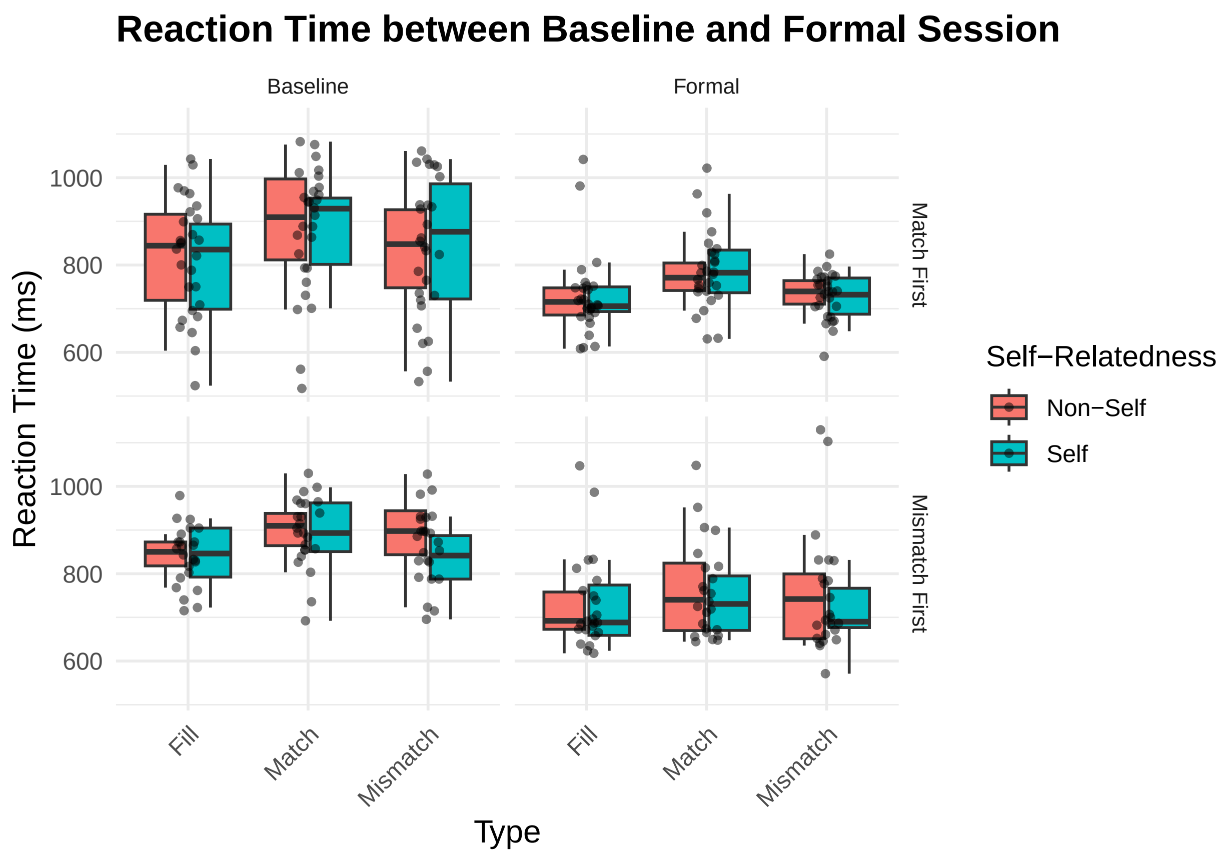
With all data ready, we first take a look at the density distribution of the average reaction time for self-related trials and non-self related trials across the Baseline and Formal experimental sessions. Overall the reaction time data follows roughly a normal distribution with Baseline skewing slightly to the left and formal slightly skewing to the right. We also observed a slightly stronger variations at the Baseline and less variations in distributions during the Formal sessions.

```r
p = ggplot(filtered_data, aes(x=meanRT, color=ifself_related)) +
  geom_density() +
  facet_grid(cols = vars(session)) +
  theme_minimal() +
  labs(
    title = "Density Distribution of Reaction Time between Baseline and Formal Session",
    x = "Reaction Time (ms)",
    y = "Frequency",
    color = "Self-Relatedness"
  )

p
```



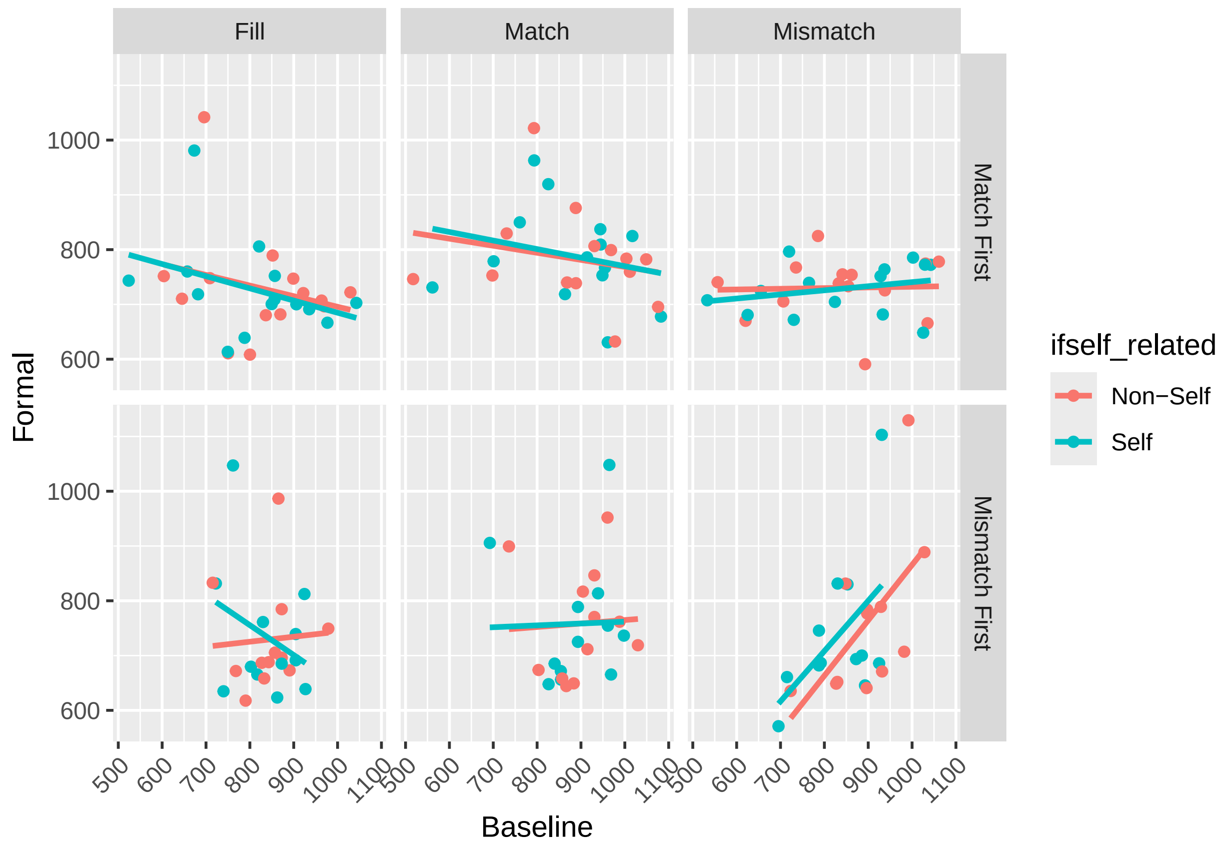
By looking at the box plot for reaction time between "Baseline" and "Formal", interestingly, we observed improvements across all experiment conditions after the training sessions. However, it is still a bit hard to tell if there is a difference in the improvement in reaction times across by self-relatedness.

```r
plot_RT <- ggplot(filtered_data, aes(x = type, y = meanRT, fill = ifself_related)) +
  geom_boxplot(outlier.shape = NA) +  # Hide outliers to clean up the plot
  geom_jitter(width = 0.1, height = 0, size = 1, alpha = 0.5, color = "black") +
  # Add jitter to show data spread
  facet_grid(group ~ session, scales = "free_x") +
  # Allow each facet to have its own x-axis scale
  theme_minimal() +
  theme(
    axis.text.x = element_text(angle = 45, hjust = 1),
    # Rotate x-axis labels to avoid overlap
    strip.text.x = element_text(size = 8),  # Smaller facet labels if needed
    strip.text.y = element_text(size = 8),
    axis.title = element_text(size = 12),  # Larger axis titles
    plot.title = element_text(size = 14, face = "bold")# Bolder and bigger title
  ) +
  labs(
    title = "Reaction Time between Baseline and Formal Session",
    x = "Type",
    y = "Reaction Time (ms)",
    fill = "Self-Relatedness"
  )

plot_RT
```



Before exploring the modeling route, let's take a look at one more plot, taking a look at the relationship between baseline and formal under each experiment conditions. Interestingly, we did not observe a stronger improvement for non-self related reaction time when compared to the self related. For the mismatch first subgroup, we actually observed a longer reaction time after all the trainings.

```
filtered_data_2 = filtered_data %>%
  select(-acc_c,-acc_all) %>%
  pivot_wider(names_from = session, values_from = meanRT)

p = ggplot(filtered_data_2, aes(x=Baseline, y=Formal, color = ifself_related)) +
  geom_point(outlier.shape = NA) +
  geom_smooth(method=lm, se=FALSE)+
  facet_grid(group ~ type)+
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

p
```



Prior Predictive Simulation & Stat. Model

To better understand and tune the priors for modeling, we conducted some online research. From an article published in 2022, Self-prioritization effect in children and adults by Singh & Karnick, we learned that self reaction time for adult aged 18-22 is round 750ms and 950ms for strangers for matched pairs. For non-matched pairs, the reaction time for the same age group is around 900ms for self and 1000ms for strangers. With this knowledge, a understanding of the data from the descriptive stats in terms of relationship between the Baseline and Formal, we can form our priors for below model.

- $Y_i|\beta_0,\beta_1,\beta_2\sigma \sim N(\mu_i,\, \sigma^2)$ with $\mu_i{=}\beta_0{+}\beta_1 conds{+}\beta_2 session$
- $\beta_0 \sim N(m_0,\, s_0^2)$ - normal distribution as observed in raw data, and we want to set mean round 800 as a result of our literature review

- $\beta_1 \sim N(m_1, s_1^2)$ - not sure about distribution but can be positive or negative
- $\sigma \sim \text{Exp}(l)$ - non-negative

Beta Prior Tuning

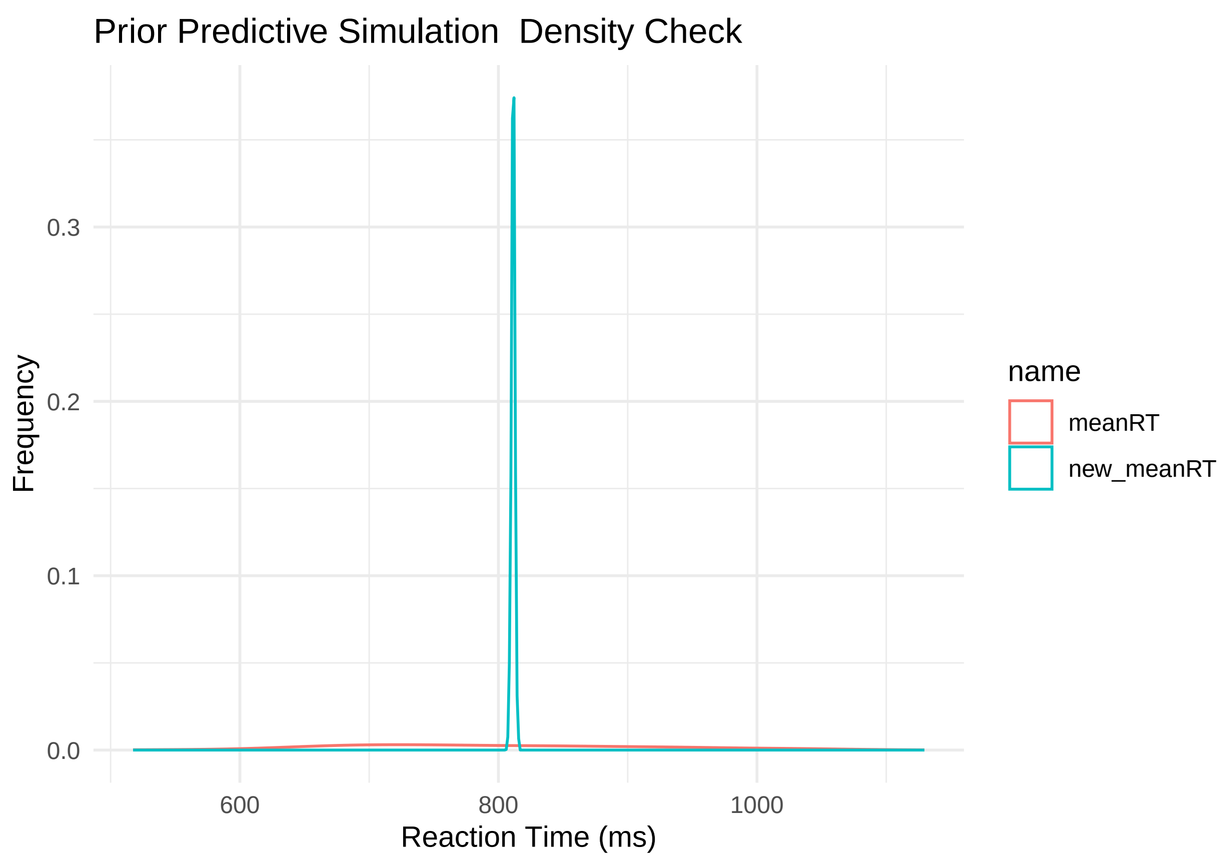
Tuning Trials	Means	Standard Deviations	Ranges	Brief Rationales
Beta0 - Trial 1	800	90	(481, 1059)	A bit smaller from the range observed in density plot
Beta0 - Trial 2	850	100	(510, 1179)	Better range and still reasonable mean based on literature review
Beta1 - Trial 1	-1	1	(-4, 2)	We observed difference slopes under difference conditions

```r
#beta prior trials
beta0_1 = rnorm(1000,800,90)
range(beta0_1)
mean(beta0_1)

beta0_2 = rnorm(1000,850,100)
range(beta0_2)
mean(beta0_2)

beta1_1 = rnorm(1000,-1,1)
range(beta1_1)
mean(beta1_1)
```

Using the prior we tuned above for prior predictive simulation, we got a very different distribution when compared to collected data. We believe this might be because the simulated data show less variations across different experiment conditions and centered a lot more around the mean.

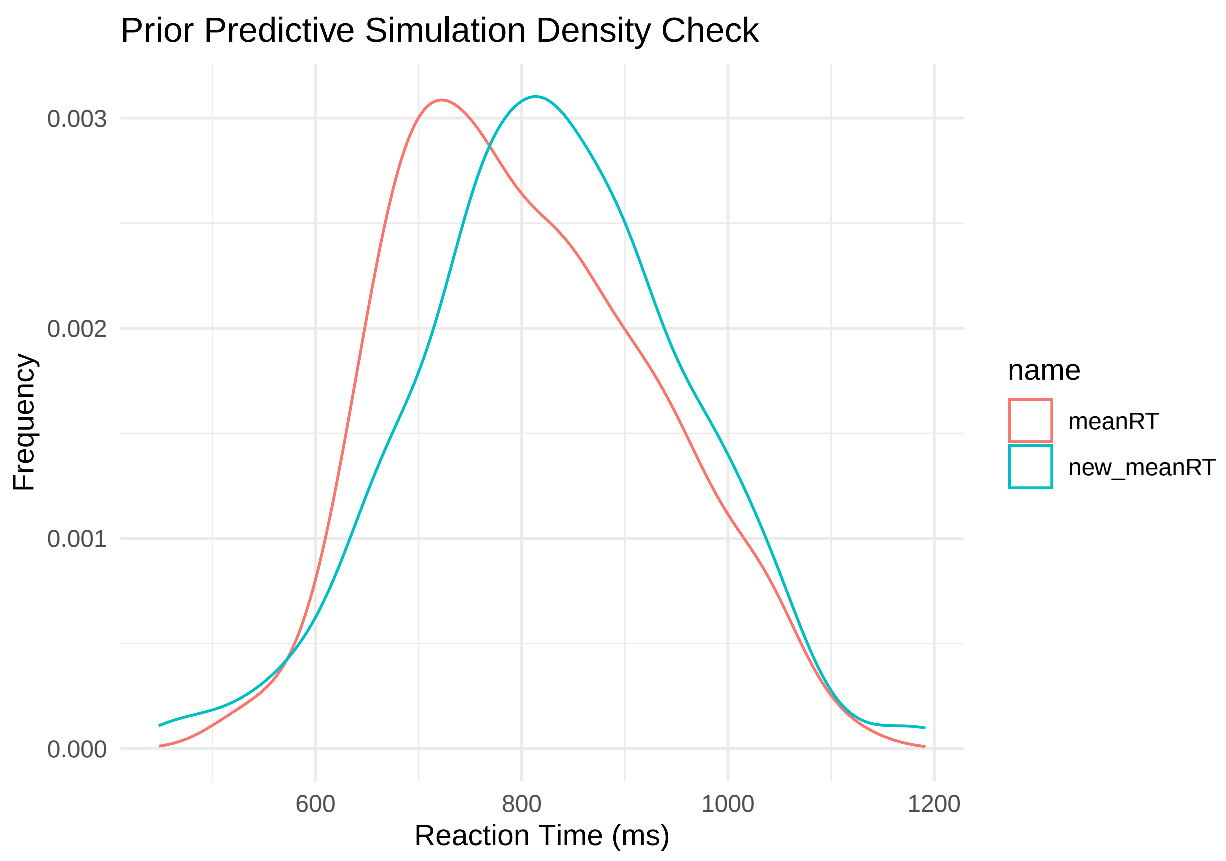
```r
#fitting the prior model for simulation
fit_1 = stan_glm(meanRT~conds+session, data = filtered_data,
                 family = gaussian,
                 prior_intercept = normal(850, 100),
                 prior = normal(-1,1),
                 prior_aux = exponential(1),
                 chains = 4, iter = 5000, prior_PD = TRUE, refresh = 0)

#getting the parameters for simulation
RT_model_df = as.data.frame(fit_1)
first_set = head(RT_model_df, 1)
beta0 = first_set$`(Intercept)`
beta1 = first_set$conds
beta2 = first_set$sessionFormal
sigma = first_set$sigma
set.seed(84735)
```

```
##Prior Predictive Simulation
filtered_data = filtered_data %>%
  mutate(session1 = as.numeric(ifelse(session=='Baseline',0,1)))
Prior_Predictive_Simulation = filtered_data %>%
  mutate(mu = beta0 + beta1 * conds + beta2 * session1,
         new_meanRT = rnorm(312, mean = mu, sd = sigma)) %>%
  select(conds, meanRT, new_meanRT) %>%
  pivot_longer(cols = meanRT:new_meanRT)

p = ggplot(Prior_Predictive_Simulation, aes(x=value, color=name)) +
  geom_density() +
  theme_minimal() +
  labs(
    title = "Prior Predictive Simulation  Density Check",
    x = "Reaction Time (ms)",
    y = "Frequency",
  )

p
```



After the first round of tuning, we decided to lower the parameter of $\sigma \sim \text{Exp}(l)$ to get a less concentrated distributions.

```
#fitting the prior model for simulation
fit_1 = stan_glm(meanRT~conds+session, data = filtered_data,
                  family = gaussian,
                  prior_intercept = normal(850, 100),
                  prior = normal(-1,1),
                  prior_aux = exponential(0.00085),
                  chains = 4, iter = 5000, prior_PD = TRUE, refresh = 0)

#getting the parameters for simulation
RT_model_df = as.data.frame(fit_1)
first_set = head(RT_model_df, 1)
beta0 = first_set$`(Intercept)`
beta1 = first_set$conds
beta2 = first_set$sessionFormal
sigma = first_set$sigma
set.seed(84735)

##Prior Predictive Simulation
filtered_data = filtered_data %>%
  mutate(session1 = as.numeric(ifelse(session=='Baseline',0,1)))
Prior_Predictive_Simulation = filtered_data %>%
  mutate(mu = beta0 + beta1 * conds + beta2 * session1,
         new_meanRT = rnorm(312, mean = mu, sd = sigma)) %>%
  select(conds, meanRT, new_meanRT) %>%
  pivot_longer(cols = meanRT:new_meanRT)

p = ggplot(Prior_Predictive_Simulation, aes(x=value, color=name)) +
  geom_density() +
  theme_minimal() +
  labs(
    title = "Prior Predictive Simulation Density Check",
    x = "Reaction Time (ms)",
    y = "Frequency",
  )

p
```



Model Fitting, PPCs, & Model Selection

For model fitting, we are thinking about a couple models including linear regression model and multi-level models. For both model, we think it's reasonable model to fit as the data follows a roughly normal distribution and the multi-level model would allows us to examine data in a more reasonable way according to the data collection process.

Linear Model For this model, we simply follow the same process in our prior predictive simulation and remove the 'prior_PD = TRUE' argument in the code.

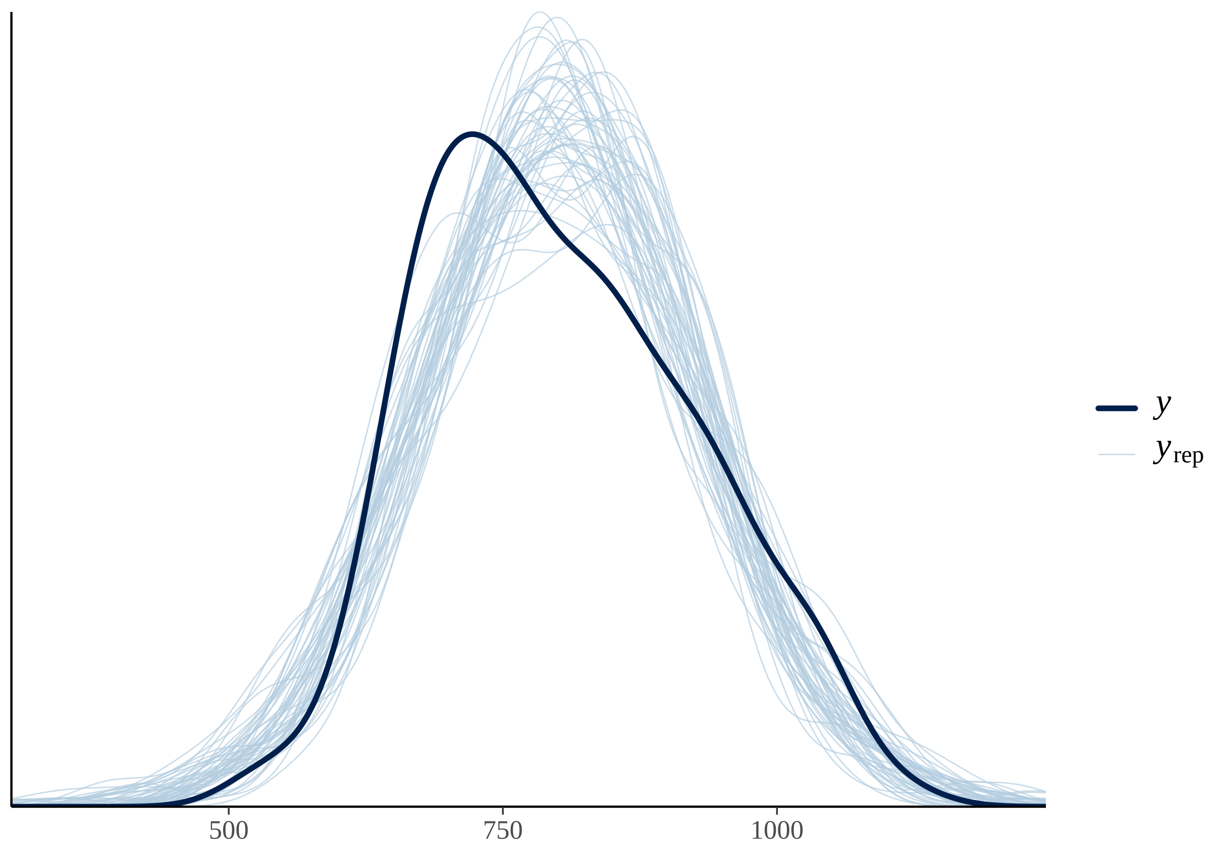
```r
set.seed(84735)
source('prediction_summary.R')
fit_2 = stan_glm(meanRT~as.factor(conds)+session, data = filtered_data,
                 family = gaussian,
                 prior_intercept = normal(850, 100),
                 prior = normal(-1,1),
                 prior_aux = exponential(0.00085),
                 chains = 4, iter = 10000, refresh = 0)
```

When checking the conducting the posterior predictive checks, we observed that:

- mae: The median absolute error, 93, measures the typical difference between reaction time and the their posterior predictive reaction time.

- mae_scaled: The scaled median absolute error, 0.7562155, measures the typical number of standard deviations that the observed reaction time fall from their posterior predictive reaction time.
- within_50: Roughly 45% of observed reaction time that fall within their 50% posterior prediction interval.
- within_95: Roughly 96% of observed reaction time that fall within their 95% posterior prediction interval.

```
pp_check(fit_2)
```



```
linear_stat = prediction_summary(fit_2, data = filtered_data)
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use ‘all_of()‘ or ‘any_of()‘ instead.
##   # Was:
##   data %>% select(y_name)
##
##   # Now:
##   data %>% select(all_of(y_name))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

For the coefficients, we care about if there is a improvement in non-self related reaction times, therefore, let's take a look at the coefficient for those conditions (4-6). Overall, there is a we observe a negative relationship on reaction time with all else remained constant and compared to the reference category of 'Fill Self'. However, this relationship is not creditable at a 80% interval as the sign changes towards the tail of the distributions for all these coefficients.

Estimates	Mean	SD	10%	50%	90%
Intercept	803	7	794	803	812
2 = Mismatch Self	-1.1	1	-2.3	-1.1	0.2
3 = Match Self	-0.9	1	-2.2	-0.9	0.4
4 = Fill Non-Self	-1.1	1	-2.4	-1.1	0.2
5 = Mismatch Non-Self	-1	1	-2.3	-1	0.3
6 = Match Non-Self	-0.9	1	-2.2	-0.9	0.4
Session = Formal	-1.5	1	-2.8	-1.5	-0.3
Sigma	123.3	5	117.1	123.1	129.8

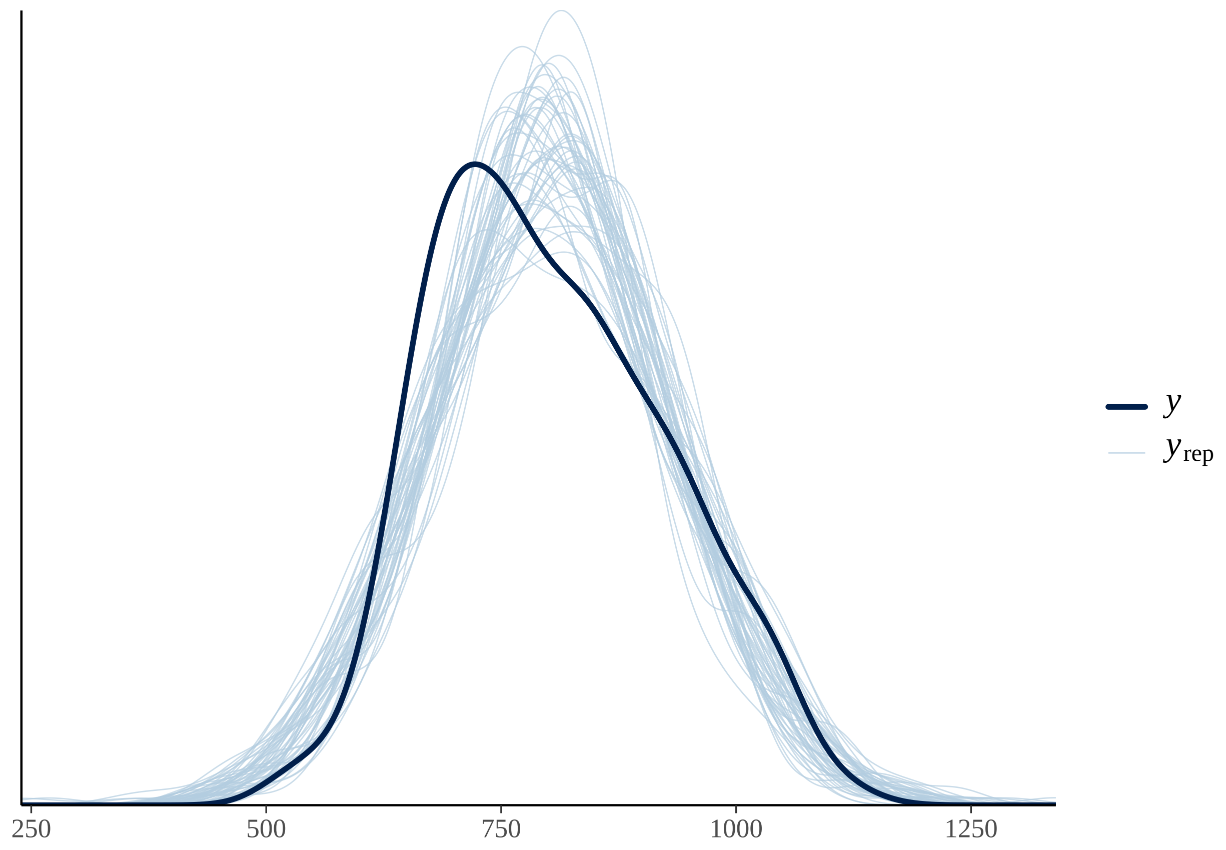
```r
summary(fit_2)
```

Multi-level Models For this model, we used the brms package

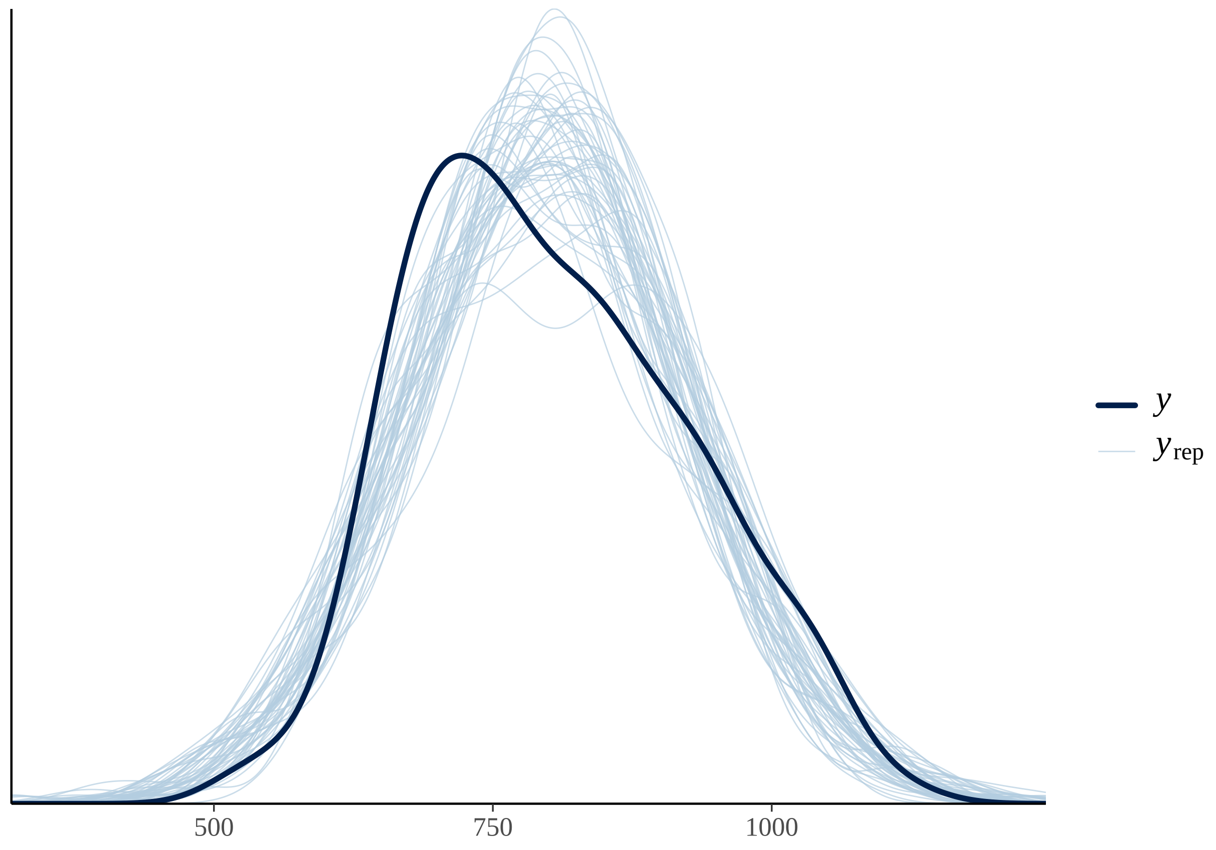
```r
model = stan_lmer(meanRT~type + ifself_related + group + session + gender + (1|subj),
                  data = filtered_data,
                  prior_intercept = normal(850, 100),
                  prior = normal(-1,1),
                  prior_aux = exponential(0.00085),
                  chains = 4, iter = 10000, refresh = 0)

fit_without_gender = stan_lmer(meanRT~type + ifself_related + group + session + (1|subj),
                               data = filtered_data,
                               prior_intercept = normal(850, 100),
                               prior = normal(-1,1),
                               prior_aux = exponential(0.00085),
                               chains = 4, iter = 10000, refresh = 0)

fit_without_self = stan_lmer(meanRT~type + gender + group + session + (1|subj),
                               data = filtered_data,
                               prior_intercept = normal(850, 100),
                               prior = normal(-1,1),
                               prior_aux = exponential(0.00085),
                               chains = 4, iter = 10000, refresh = 0)

fit_without_gender_self = stan_lmer(meanRT~type + group + session + (1|subj),
                               data = filtered_data,
                               prior_intercept = normal(850, 100),
                               prior = normal(-1,1),
                               prior_aux = exponential(0.00085),
                               chains = 4, iter = 10000, refresh = 0)
```

When comparing the posterior predictive plots, all of the models we fitted with multi-level model did not deviates too much from the linear posterior predictive plots.

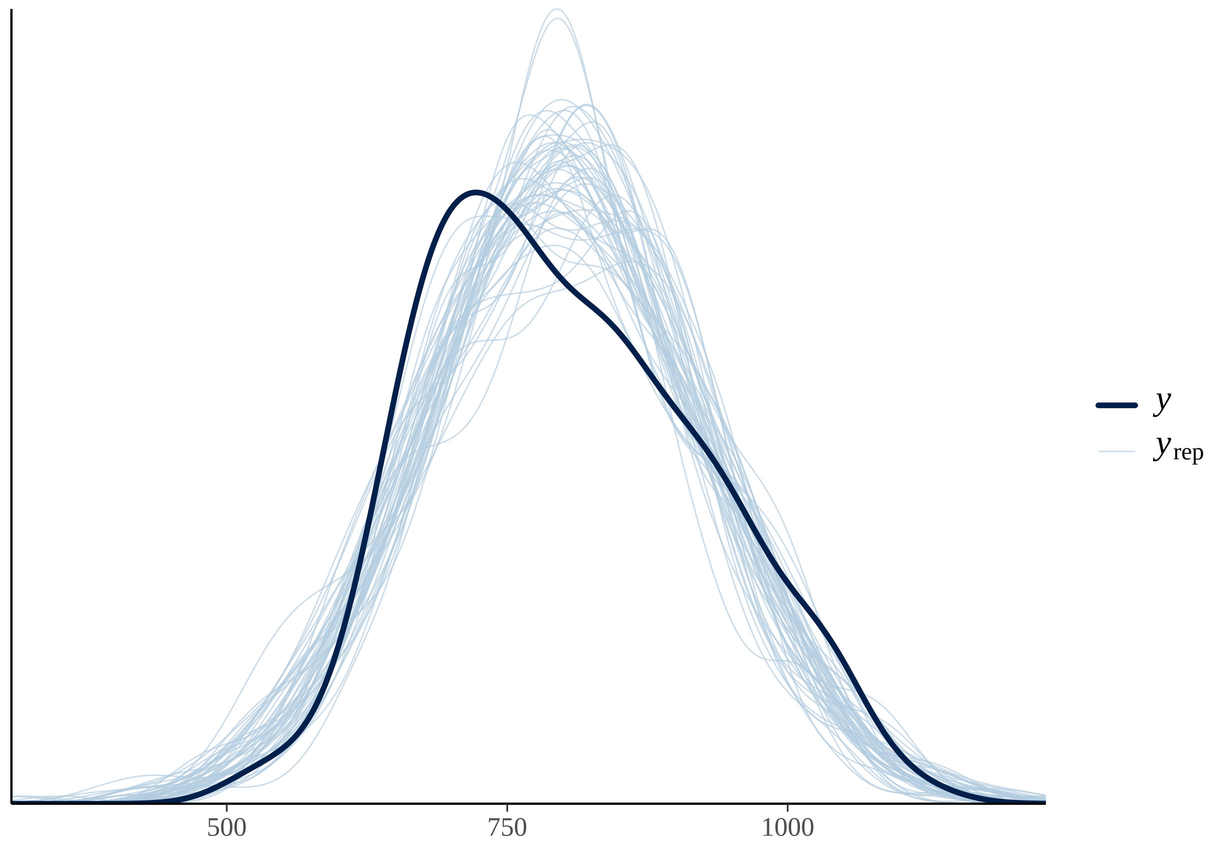
```r
par(mfrow=c(2,2))
pp_check(model)
```



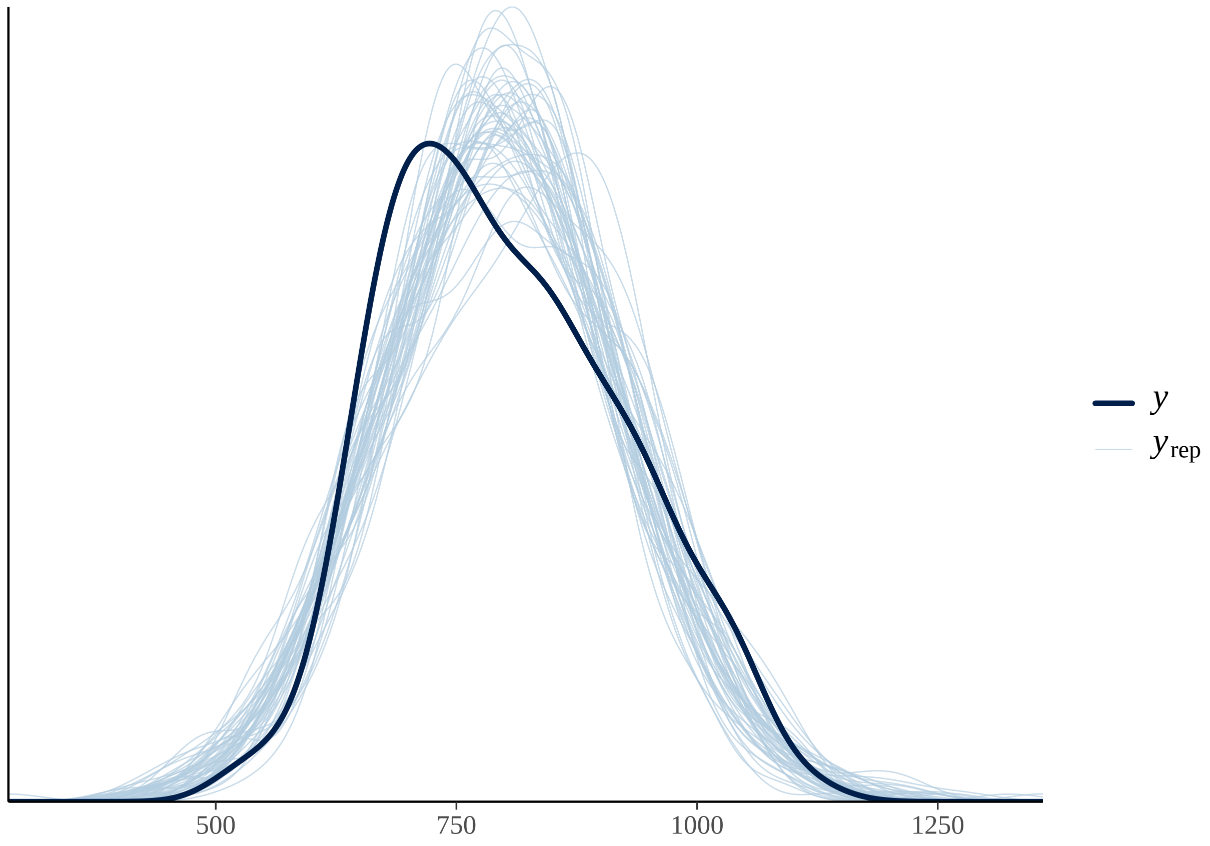
```r
pp_check(fit_without_gender)
```



```
pp_check(fit_without_self)
```



```r
pp_check(fit_without_gender_self)
```



When comparing the mean absolute error across all fitted models, the linear model actually shows higher error then the multilevel models, and less collected data fall within the 95% interval of the posterior linear model.

```
all_predictor = prediction_summary(model, data = filtered_data)
without_gender = prediction_summary(fit_without_gender, data = filtered_data)
without_self = prediction_summary(fit_without_self, data = filtered_data)
without_gender_self = prediction_summary(fit_without_gender_self, data = filtered_data)
table = rbind(linear_stat,all_predictor,without_gender,without_self,without_gender_self)
rowname = c("Linear", "Multi-level: all_predictor", "Multi-level: without_gender",
            "Multi-level: without_self","Multi-level: without_gender_self")
cbind(rowname,table)
```

```
##                             rowname      mae mae_scaled within_50 within_95
## 1                            Linear 93.45344  0.7562155 0.4455128 0.9615385
## 2        Multi-level: all_predictor 86.59248  0.7748883 0.4166667 0.9871795
## 3       Multi-level: without_gender 87.25298  0.7785118 0.4198718 0.9839744
## 4         Multi-level: without_self 87.08784  0.7766668 0.4294872 0.9871795
## 5 Multi-level: without_gender_self 87.25252  0.7773847 0.4166667 0.9871795
```

When comparing the model using LOO, we pay attention to elpd_diff values, which are differences in expected log predictive densities, where higher values indicate better predictive performance. In our case, it is the loo_without_self model. However, since self is an important predictor to our experiment, we actually going to move forward with the one with all predictor variables.

```
loo_linear <- loo(fit_2)
loo_all <- loo(model)
loo_without_gender <- loo(fit_without_gender)
loo_without_self <- loo(fit_without_self)
loo_without_gender_self <- loo(fit_without_gender_self)
model_comparison <- loo_compare(loo_linear,loo_all,loo_without_gender,
                                loo_without_self,loo_without_gender_self)
model_comparison
```

```
##                         elpd_diff se_diff
## fit_without_gender_self   0.0       0.0
## model                     0.0       0.1
## fit_without_gender        0.0       0.1
## fit_without_self         -0.2       0.1
## fit_2                   -30.1       7.5
```

Similar to the linear model, we only observed in significant change in reaction time for Baseline vs Formal. This means that when everything else remained constant, the same individual in Baseline session would show a longer reaction time than the same individual in the formal session. When looking at individual level grouping, we observed inconsistent results across the individual for within effects. This could be due to the limited experiment units within each groups at this level.

```
Final_model_stats = data.frame(model$stan_summary)
Final_model_stats = Final_model_stats %>%
  select(mean,sd,X10.,X90.,n_eff,Rhat) %>%
  rename("10%" = "X10.", "90%" = "X90.")
Final_model_stats
```

```
##                                         mean         sd         10%
## (Intercept)                     8.050331e+02 13.7897436  787.705535
## typeMatch                      -7.464325e-01  0.9875415   -2.018123
## typeMismatch                   -1.055017e+00  1.0001633   -2.331684
## ifself_relatedSelf             -1.065350e+00  1.0004350   -2.339078
## groupMismatch First            -9.852982e-01  0.9875469   -2.259339
## sessionFormal                  -1.702749e+00  1.0046978   -3.004843
## gender2                        -9.960116e-01  0.9934171   -2.259465
## b[(Intercept) subj:1]           1.107830e+00 29.5730718  -37.088418
## b[(Intercept) subj:2]          -3.795295e+01 30.0503797  -76.895680
## b[(Intercept) subj:3]           4.615140e+01 30.2278383    7.239966
## b[(Intercept) subj:4]          -5.501214e+00 29.7470809  -43.652808
## b[(Intercept) subj:5]          -3.289284e+01 29.5897182  -70.594276
## b[(Intercept) subj:6]          -7.962340e+01 30.4678734 -118.614734
## b[(Intercept) subj:7]           5.732295e+01 30.2992821   18.652033
## b[(Intercept) subj:8]          -1.257217e+02 31.5463681 -166.089887
## b[(Intercept) subj:9]           2.010621e+01 29.8219101  -17.680063
## b[(Intercept) subj:10]          5.046264e+01 30.1587907   12.001124
## b[(Intercept) subj:11]          2.640295e+01 29.6840120  -11.606471
## b[(Intercept) subj:12]          2.285095e+01 29.9277946  -15.333999
## b[(Intercept) subj:13]          6.890721e-02 29.7858204  -37.979758
## b[(Intercept) subj:14]         -4.613069e+01 30.0715453  -84.635395
## b[(Intercept) subj:15]          1.023902e+01 29.8527874  -27.847305
## b[(Intercept) subj:16]          4.712059e+00 29.6945169  -33.154673
## b[(Intercept) subj:17]              5.764881e+01   30.0647042    19.502860
## b[(Intercept) subj:18]              1.382856e+02   31.6899844    98.140422
## b[(Intercept) subj:19]             -5.144912e+01   30.5106280   -90.436478
## b[(Intercept) subj:20]             -2.398111e+01   29.9787164   -61.977702
## b[(Intercept) subj:21]             -4.749596e+01   29.9248634   -86.045886
## b[(Intercept) subj:22]             -4.662110e+01   30.6416377   -85.752423
## b[(Intercept) subj:23]              1.839334e+01   29.8079366   -19.842730
## b[(Intercept) subj:24]              4.208452e+01   30.0425950     4.440126
## b[(Intercept) subj:25]             -3.367352e+01   30.1971322   -72.727232
## b[(Intercept) subj:26]              1.367087e+01   29.7171205   -24.086185
## b[(Intercept) subj:_NEW_subj]       3.738737e-02   63.2380189   -79.367128
## sigma                               1.082345e+02    4.5750442   102.514130
## Sigma[subj:(Intercept),(Intercept)]  3.945673e+03 1533.7236877  2305.753610
## mean_PPD                            8.015334e+02    8.6511815   790.413841
## log-posterior                      -1.959901e+03    5.6673519 -1967.323463
##                                              90%     n_eff      Rhat
## (Intercept)                          822.2725843  5892.629 1.0007143
## typeMatch                              0.5139512 33908.227 0.9998775
## typeMismatch                           0.2178543 31210.244 1.0000654
## ifself_relatedSelf                     0.2134774 32534.317 0.9999658
## groupMismatch First                    0.2792096 31674.462 1.0000286
## sessionFormal                         -0.4158627 32094.988 0.9999868
## gender2                                0.2703179 32978.895 1.0000587
## b[(Intercept) subj:1]                 38.5571720 18447.519 1.0001157
## b[(Intercept) subj:2]                  0.1160512 19218.677 1.0001722
## b[(Intercept) subj:3]                 84.6835867 18660.480 1.0000161
## b[(Intercept) subj:4]                 32.7804720 18085.049 1.0000701
## b[(Intercept) subj:5]                  4.7821590 18436.431 1.0002958
## b[(Intercept) subj:6]                -41.2076271 18006.012 1.0001703
## b[(Intercept) subj:7]                 96.0824943 18023.858 1.0000313
## b[(Intercept) subj:8]                -85.9888831 15913.225 1.0003301
## b[(Intercept) subj:9]                 57.9655720 18873.949 1.0001306
## b[(Intercept) subj:10]                89.1086949 18398.910 1.0001504
## b[(Intercept) subj:11]                64.4596897 18264.147 0.9999877
## b[(Intercept) subj:12]                60.9701045 18627.285 1.0000428
## b[(Intercept) subj:13]                38.1286320 19177.005 1.0001005
## b[(Intercept) subj:14]                -7.7729469 17178.621 1.0001514
## b[(Intercept) subj:15]                48.3344706 18486.791 0.9999264
## b[(Intercept) subj:16]                42.5316527 18534.283 0.9999395
## b[(Intercept) subj:17]                96.2196337 19327.121 0.9998701
## b[(Intercept) subj:18]               178.8701505 17308.380 0.9998709
## b[(Intercept) subj:19]               -12.4247073 17698.138 0.9999356
## b[(Intercept) subj:20]                14.3769485 18021.009 1.0000794
## b[(Intercept) subj:21]                -9.8560743 18235.393 1.0000305
## b[(Intercept) subj:22]                -7.3270812 18354.404 1.0004364
## b[(Intercept) subj:23]                56.7001157 18814.288 0.9999191
## b[(Intercept) subj:24]                80.6479316 18838.687 1.0001551
## b[(Intercept) subj:25]                 4.4503541 18749.477 1.0000747
## b[(Intercept) subj:26]                51.7027358 18566.410 0.9999389
## b[(Intercept) subj:_NEW_subj]         78.4397848 31987.648 0.9999147
## sigma                                114.1760061 24637.366 0.9999478
## Sigma[subj:(Intercept),(Intercept)]  5927.6479278  7145.998 1.0000354
## mean_PPD                             812.6612157 19964.812 0.9998746
## log-posterior                      -1952.8945890  4278.975 1.0016797
```

Discussion

For this analysis, we were only able to find significant change in reaction time in terms of experiment sessions, Baseline vs Formal. This means that overall training is helpful at reduction reaction times to a prompt regardless if it's "self-related". We can't really prove that if the reverse training help reduce self-prioritization or not as the coefficient for "self vs non-self" is in the consistent direction with in the 80% creditable interval.

Some of the limitations, we faced with this analysis is the limited time. We did not have enough time to think through the data carefully to account for accuracy and reaction time together.

Attribution

Raw Data Summary

- Chenghao worked on the data cleaning. We worked together to plot the raw data summary as there were a lot discussion around how we can make sense of the data given the complicated experimental design.

Prior Predictive Simulation

- Xunqing worked on the prior predictive simulation and tuning of the prior

Modeling Fitting, PPCs, and Model Selection

- We will both be working on this part of the project, where Chenghao fitted all the multi-level models, key convergence diagnostics, including R hats. Xunqing worked on the linear fit.

Discussion

- We worked on this part together

Citation

Singh, D., & Karnick, H. (2022, May 24). Self-prioritization effect in children and adults. Frontiers. https://www.frontiersin.org/journals/psychology/articles/10.3389/fpsyg.2022.726230/full