EM2_databehandling

2025-12-03

Pre-processing of data

```
# get list of data files in current directory (excluding questionnaire data)
list_of_files <- setdiff(dir(pattern=".csv"), "questionnaire.csv")

# create data frame
df <- read_delim(list_of_files) %>%
  mutate(ReactionTime = ReactionTime * 1000)
```

```
## Rows: 2640 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (6): subject_id, subject_name, Condition, Agent_Type, response, X_Hit
## dbl (4): trial_num, ReactionTime, X_RT, X_FalseAlarm_Count
## lgl (3): Ball_present, Part_belief, Agent_belief
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# identify subjects to exclude
slow_subjects <- df %>%
  filter(Ball_present == TRUE) %>%
  group_by(subject_id) %>%
  summarise(RT_over_1000 = sum(ReactionTime > 1000, na.rm = TRUE)) %>%
  filter(RT_over_1000 > 4) %>%
  pull(subject_id)

error_subjects <- df %>%
  group_by(subject_id) %>%
  summarise(error_trials = sum(response == "missed" |
              response == "false_alarm" |
              (X_Hit == "No" & response == "hit"))) %>%
  filter(error_trials > 8) %>%
  pull(subject_id)

# exclude subjects
df_new <- df %>%
  filter(!subject_id %in% slow_subjects, !subject_id %in% error_subjects)

# exclude trials (slow RTs and no X-hit)
df_ex <- df_new %>%
  filter(!(Ball_present == TRUE & response == "hit" & ReactionTime > 1000),
         X_Hit == "Yes")

# create final data frame to use for analysis
```

```r
df_final <- df_ex %>%
  filter(Ball_present == TRUE, response == "hit") %>%
  group_by(subject_id) %>%
  mutate(Agent_First  = first(Agent_Type),
         Agent_Second = if_else(Agent_First == "Self", "Smurf", "Self")
         ) %>%
  ungroup()
```

Pre-processing of questionnaire data

```r
# create questionnaire data frame
quest <- read.csv("questionnaire.csv", sep=";")

# correct subject-id's to fit other data frame
quest$subject_id[quest$subject_id == 1] <- "0001"
quest$subject_id[quest$subject_id == 409] <- "0409"
quest$subject_id[quest$subject_id == 511] <- "0511"
quest$subject_id[quest$subject_id == 708] <- "0708"

# compute immersiveness scores
quest_scores <- quest %>%
  mutate(score_smurf = (
           Q1_smurf +
           Q2_smurf +
           Q3_smurf +
           Q4_smurf +
           Q5_smurf +
           Q6_smurf +
           Q7_smurf +
           Q8_smurf) / 8,
         score_self = (
           Q1_self +
           Q2_self +
           Q3_self +
           Q4_self +
           Q5_self +
           Q6_self +
           Q7_self +
           Q8_self) / 8
         )

# combine data frames
df_combined <- df_final %>%
  group_by(subject_id, Agent_Type, Condition, Agent_First) %>%
  summarise(meanRT = mean(ReactionTime, na.rm = TRUE)) %>%
  ungroup() %>%
  left_join(quest_scores %>% select(subject_id, score_smurf, score_self, Q1_self, Q8_self),
            by = "subject_id")
```

```
## `summarise()` has grouped output by 'subject_id', 'Agent_Type', 'Condition'.
## You can override using the `.groups` argument.
```

Accuracy

```
no_x_hit <- df_new %>%
  filter(X_Hit == "No")

slow_trials <- df_new %>%
  filter(Ball_present == TRUE & response == "hit" & ReactionTime > 1000)

ex_trials <- df_new %>%
  filter(X_Hit == "No" |
         (Ball_present == TRUE & response == "hit" & ReactionTime > 1000))

hits <- df_ex %>%
  filter(response == "hit")

misses <- df_ex %>%
  filter(response == "missed")

false_alarms <- df_ex %>%
  filter(response == "false_alarm")

correct_rejections <- df_ex %>%
  filter(response == "no_press")

# calculate errors
errors <- nrow(false_alarms) + nrow(misses)

# calculate error percentage
(errors / nrow(df_ex)) * 100
```

```
## [1] 2.192801
```

```
# calculate false alarm percentage
(nrow(false_alarms) / nrow(df_ex)) * 100
```

```
## [1] 1.985933
```

```
# calculate miss percentage
(nrow(misses) / nrow(df_ex)) * 100
```

```
## [1] 0.206868
```

```
# no x hit percentage
nrow(no_x_hit) / nrow(df_new) * 100
```

```
## [1] 2.177419
```

```
# slow trials percentage
nrow(slow_trials) / nrow(df_new) * 100
```

```
## [1] 0.4435484
```

```
# excluded trials percentage
nrow(ex_trials) / nrow(df_new) * 100
```

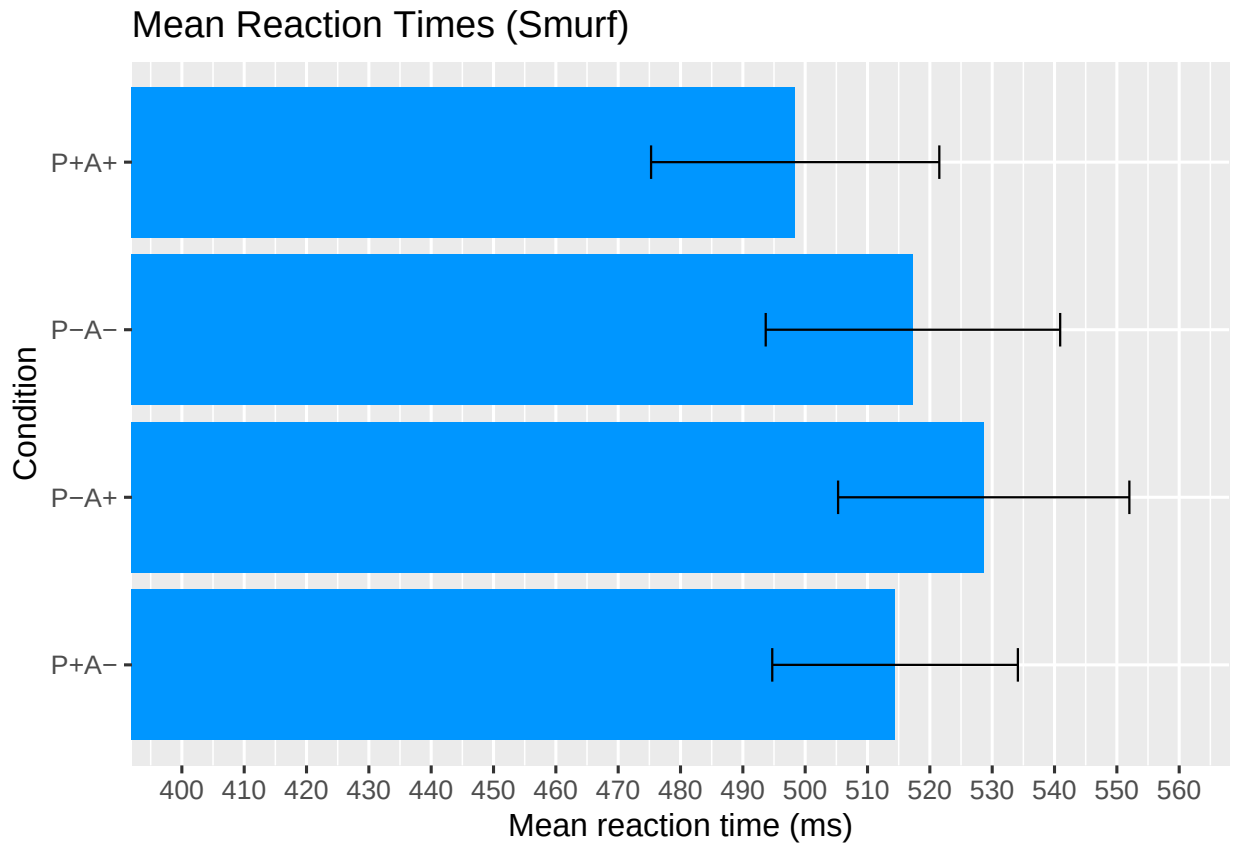
```
## [1] 2.540323
```

RT plots

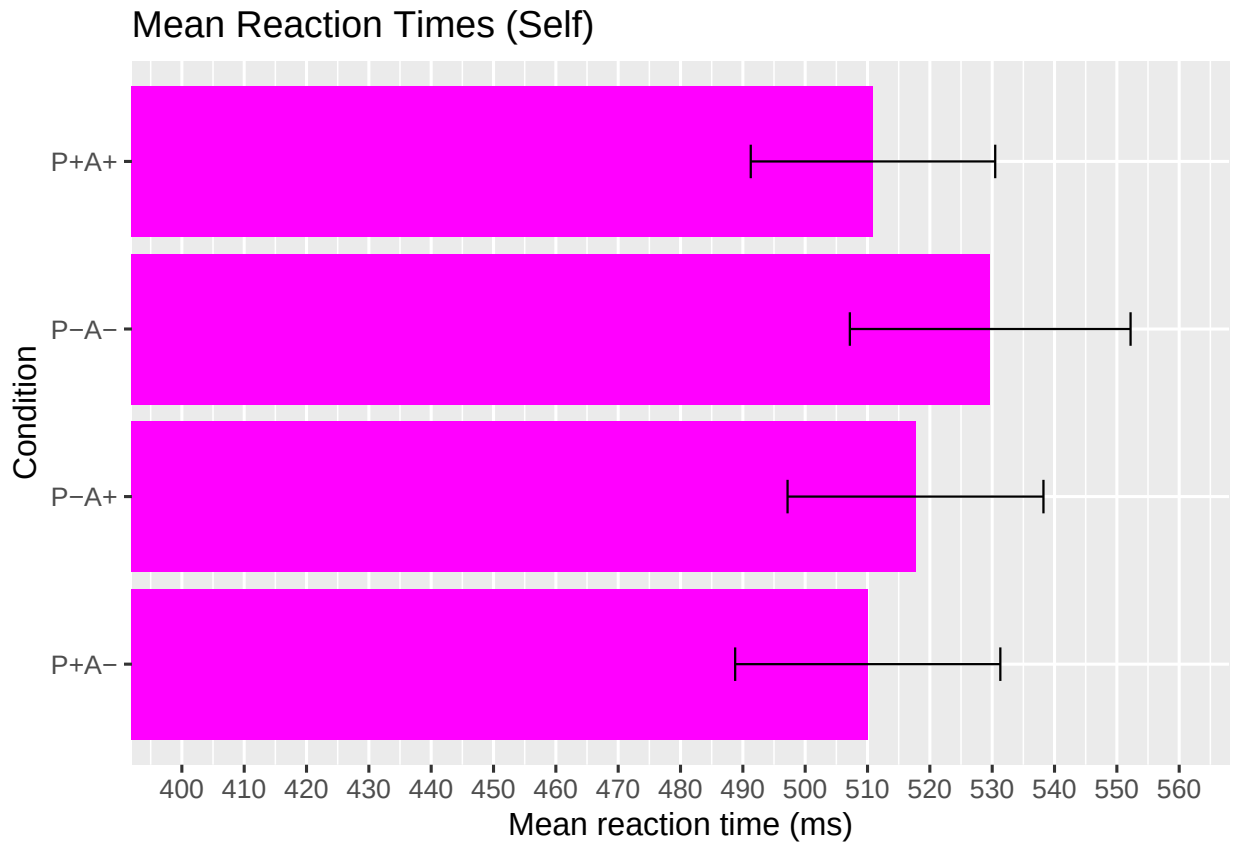
```
# compute mean RTs for smurf trials
means_smurf <- df_combined %>%
  filter(Agent_Type == "Smurf") %>%
  group_by(Condition) %>%
  summarise(meanRT_smurf = mean(meanRT, na.rm = TRUE),
            SD = sd(meanRT),
            N = n(),
            SE = SD / sqrt(N))

# compute mean RTs for self trials
means_self <- df_combined %>%
  filter(Agent_Type == "Self") %>%
  group_by(Condition) %>%
  summarise(meanRT_self = mean(meanRT, na.rm = TRUE),
            SD = sd(meanRT),
            N = n(),
            SE = SD / sqrt(N))

# compute overall mean RTs
means_overall <- df_combined %>%
  group_by(Condition) %>%
  summarise(meanRT_overall = mean(meanRT, na.rm = TRUE),
            SD = sd(meanRT),
            N = n(),
            SE = SD / sqrt(N))

# change order of conditions in data frames
means_smurf$Condition <- factor(means_smurf$Condition,
                                levels = c("P+A-","P-A+","P-A-","P+A+"))

means_self$Condition <- factor(means_self$Condition,
                               levels = c("P+A-","P-A+","P-A-","P+A+"))

means_overall$Condition <- factor(means_self$Condition,
                                  levels = c("P+A-","P-A+","P-A-","P+A+"))

# plot mean RTs for smurf trials
ggplot(means_smurf, aes(x = meanRT_smurf, y = Condition)) +
  geom_col(fill = "#0096FF") +
  geom_errorbarh(aes(xmin = meanRT_smurf - SE, xmax = meanRT_smurf + SE),
                 height = 0.2, size = 0.4) +
  coord_cartesian(xlim = c(400, 560)) +
  scale_x_continuous(
    breaks = seq(400, 560, by = 10)
    ) +
  labs(title = "Mean Reaction Times (Smurf)",
       x = "Mean reaction time (ms)",
       y = "Condition"
       ) +
  theme_grey(base_size = 12)
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
```

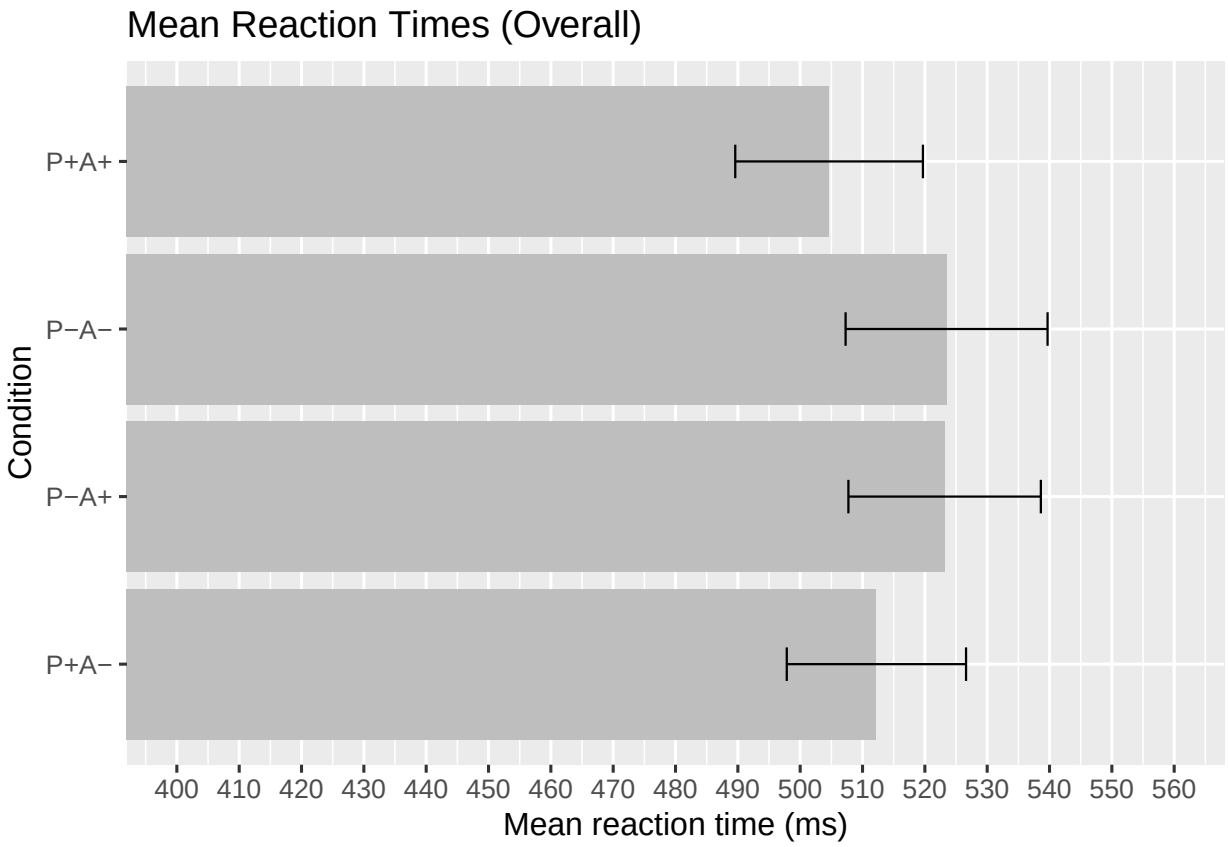
```
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




```
# plot mean RTs for self trials
ggplot(means_self, aes(x = meanRT_self, y = Condition)) +
  geom_col(fill = "#FF00FF") +
  geom_errorbarh(aes(xmin = meanRT_self - SE, xmax = meanRT_self + SE),
                 height = 0.2, size = 0.4) +
  coord_cartesian(xlim = c(400, 560)) +
  scale_x_continuous(
    breaks = seq(400, 560, by = 10)
    ) +
  labs(title = "Mean Reaction Times (Self)",
       x = "Mean reaction time (ms)",
       y = "Condition"
       ) +
  theme_grey(base_size = 12)
```




```
# plot overall mean RTs
ggplot(means_overall, aes(x = meanRT_overall, y = Condition)) +
  geom_col(fill = "grey") +
  geom_errorbarh(aes(xmin = meanRT_overall - SE, xmax = meanRT_overall + SE),
                 height = 0.2, size = 0.4) +
  coord_cartesian(xlim = c(400, 560)) +
  scale_x_continuous(
    breaks = seq(400, 560, by = 10)
    ) +
  labs(title = "Mean Reaction Times (Overall)",
       x = "Mean reaction time (ms)",
       y = "Condition"
       ) +
  theme_grey(base_size = 12)
```



T-tests for ToM-effect

```
# transform to wide format
df_wide <- df_combined %>%
  pivot_wider(
    names_from = Condition,
    values_from = meanRT
  ) %>%
  mutate(mean_diff = `P-A-` - `P-A+`)

# filter self trials
df_self_wide <- df_wide %>%
  filter(Agent_Type == "Self")

# filter smurf trials
df_smurf_wide <- df_wide %>%
  filter(Agent_Type == "Smurf")

# self ToM-effect
ttestBF(df_self_wide$`P-A+`, df_self_wide$`P-A-`, paired = TRUE)
```

```
## Bayes factor analysis
## --------------
## [1] Alt., r=0.707 : 0.3396978 ±0.03%
##
## Against denominator:
```

```
##   Null, mu = 0
## ---
## Bayes factor type: BFoneSample, JZS
```

```
# smurf ToM-effect
ttestBF(df_smurf_wide$`P-A+`, df_smurf_wide$`P-A-`, paired = TRUE)
```

```
## Bayes factor analysis
## --------------
## [1] Alt., r=0.707 : 0.2686829 ±0.03%
##
## Against denominator:
##   Null, mu = 0
## ---
## Bayes factor type: BFoneSample, JZS
```

```
# overall ToM-effect
ttestBF(df_wide$`P-A+`, df_wide$`P-A-`, paired = TRUE)
```

```
## Bayes factor analysis
## --------------
## [1] Alt., r=0.707 : 0.1391593 ±0.09%
##
## Against denominator:
##   Null, mu = 0
## ---
## Bayes factor type: BFoneSample, JZS
```

```
# difference between self and smurf in ToM-effect
ttestBF(df_self_wide$mean_diff, df_smurf_wide$mean_diff, paired = TRUE)
```

```
## Bayes factor analysis
## --------------
## [1] Alt., r=0.707 : 0.4467895 ±0.03%
##
## Against denominator:
##   Null, mu = 0
## ---
## Bayes factor type: BFoneSample, JZS
```

Block order effects

```
# filter by block order
df_self_first <- df_wide %>%
  filter(Agent_First == "Self")

df_smurf_first <- df_wide %>%
  filter(Agent_First == "Smurf")

# filter by block order and agent type
df_self_first_self <- df_self_first %>%
  filter(Agent_Type == "Self")

df_self_first_smurf <- df_self_first %>%
  filter(Agent_Type == "Smurf")
```

```r
df_smurf_first_smurf <- df_smurf_first %>%
  filter(Agent_Type == "Smurf")

df_smurf_first_self <- df_smurf_first %>%
  filter(Agent_First == "Smurf")

# self first ToM-effect (only self trials)
ttestBF(df_self_first_self$`P-A+`, df_self_first_self$`P-A-`, paired = TRUE)
```

```
## Bayes factor analysis
## --------------
## [1] Alt., r=0.707 : 0.8728702 ±0.02%
##
## Against denominator:
##   Null, mu = 0
## ---
## Bayes factor type: BFoneSample, JZS
```

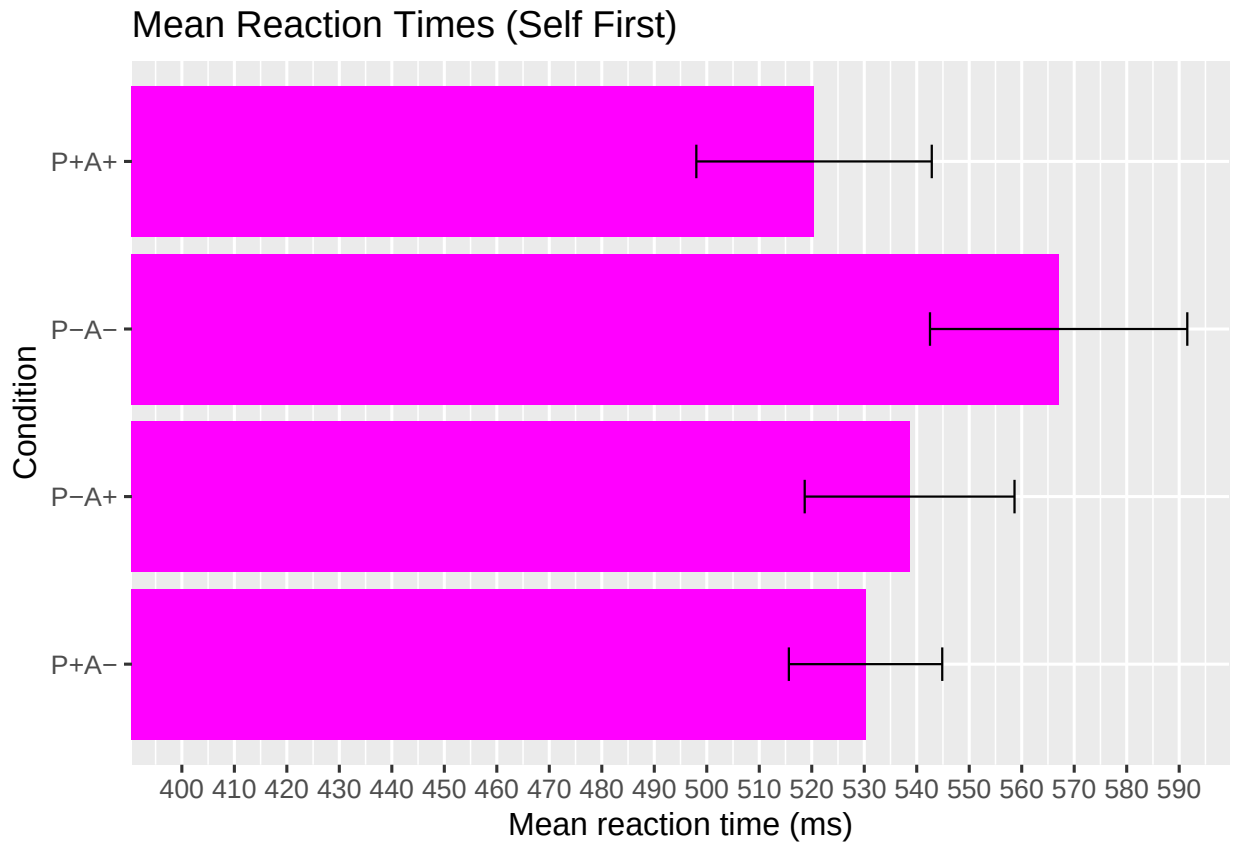
```r
# self first ToM-effect (all trials)
ttestBF(df_self_first$`P-A+`, df_self_first$`P-A-`, paired = TRUE)
```

```
## Bayes factor analysis
## --------------
## [1] Alt., r=0.707 : 0.2489915 ±0.04%
##
## Against denominator:
##   Null, mu = 0
## ---
## Bayes factor type: BFoneSample, JZS
```

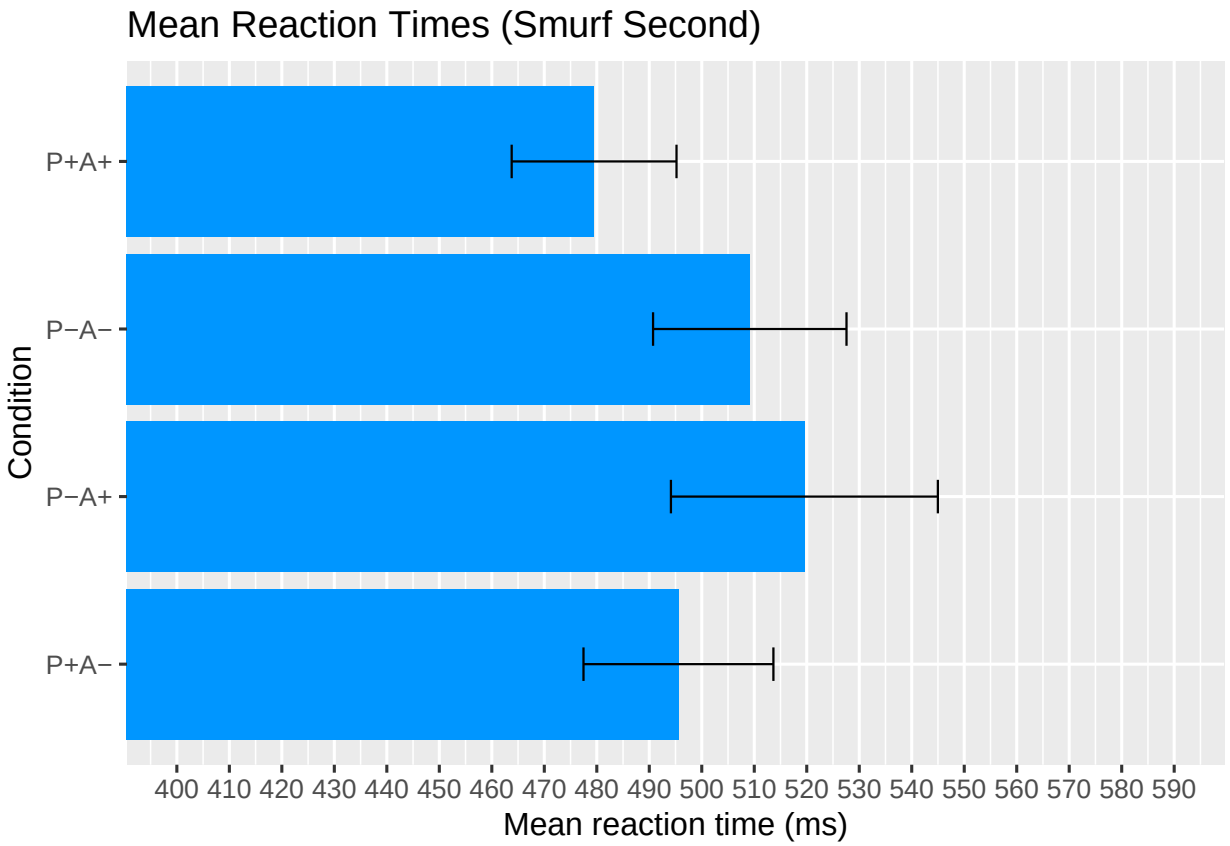
Blockwise RT plots

```r
# compute mean RTs for self trials (when self is first)
means_self_first_self <- df_combined %>%
  filter(Agent_First == "Self", Agent_Type == "Self") %>%
  group_by(Condition) %>%
  summarise(meanRT_self = mean(meanRT, na.rm = TRUE),
            SD = sd(meanRT),
            N = n(),
            SE = SD / sqrt(N))

# compute mean RTs for smurf trials (when self is first)
means_self_first_smurf <- df_combined %>%
  filter(Agent_First == "Self", Agent_Type == "Smurf") %>%
  group_by(Condition) %>%
  summarise(meanRT_smurf = mean(meanRT, na.rm = TRUE),
            SD = sd(meanRT),
            N = n(),
            SE = SD / sqrt(N))

# compute mean RTs for self trials (when smurf is first)
means_smurf_first_self <- df_combined %>%
  filter(Agent_First == "Smurf", Agent_Type == "Self") %>%
  group_by(Condition) %>%
```

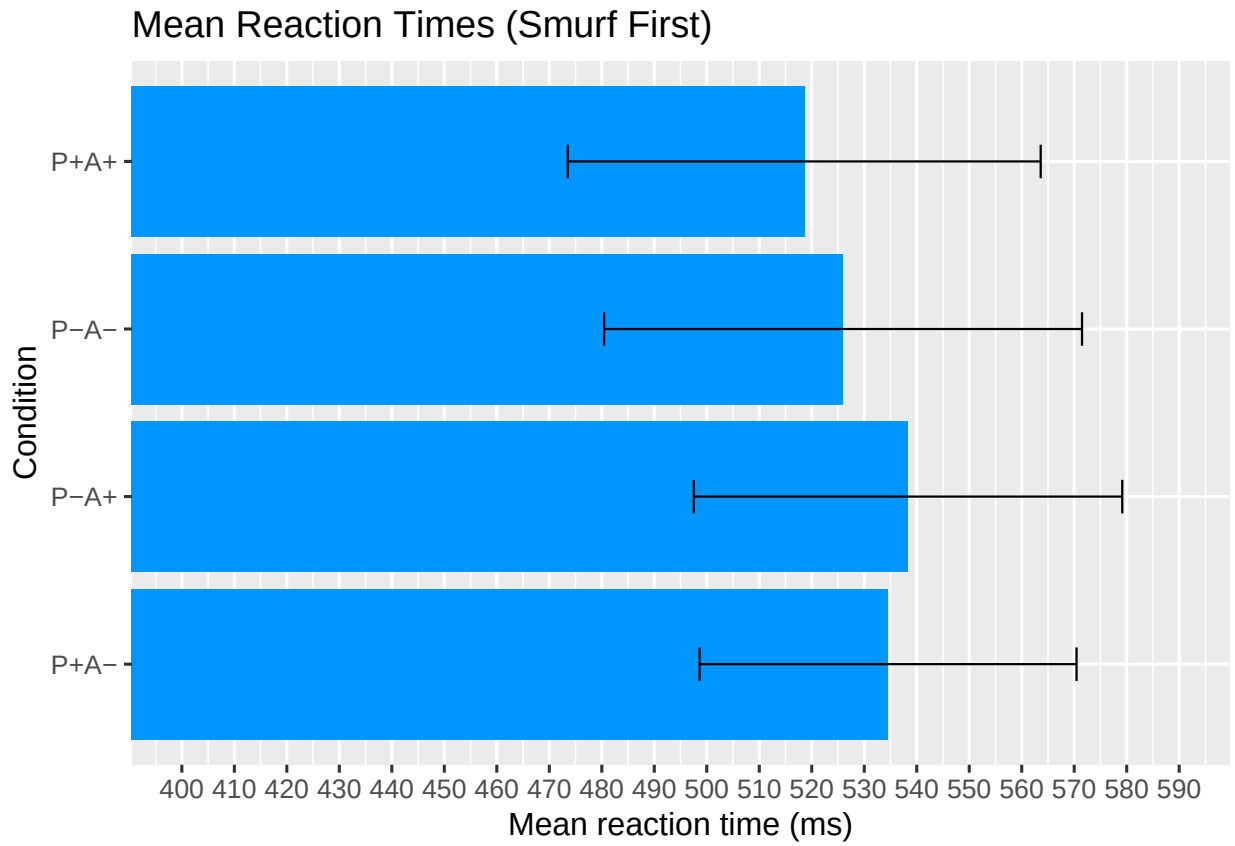
```r
  summarise(meanRT_self = mean(meanRT, na.rm = TRUE),
            SD = sd(meanRT),
            N = n(),
            SE = SD / sqrt(N))

# compute mean RTs for self trials (when smurf is first)
means_smurf_first_smurf <- df_combined %>%
  filter(Agent_First == "Smurf", Agent_Type == "Smurf") %>%
  group_by(Condition) %>%
  summarise(meanRT_smurf = mean(meanRT, na.rm = TRUE),
            SD = sd(meanRT),
            N = n(),
            SE = SD / sqrt(N))


# change order of conditions in data frame
means_self_first_self$Condition <- factor(means_self$Condition,
                                          levels = c("P+A-","P-A+","P-A-","P+A+"))

means_self_first_smurf$Condition <- factor(means_self$Condition,
                                          levels = c("P+A-","P-A+","P-A-","P+A+"))

means_smurf_first_self$Condition <- factor(means_self$Condition,
                                          levels = c("P+A-","P-A+","P-A-","P+A+"))

means_smurf_first_smurf$Condition <- factor(means_self$Condition,
                                          levels = c("P+A-","P-A+","P-A-","P+A+"))

# plot mean RTs for self in first block
ggplot(means_self_first_self, aes(x = meanRT_self, y = Condition)) +
  geom_col(fill = "#FF00FF") +
  geom_errorbarh(aes(xmin = meanRT_self - SE, xmax = meanRT_self + SE),
                 height = 0.2, size = 0.4) +
  coord_cartesian(xlim = c(400, 590)) +
  scale_x_continuous(
    breaks = seq(400, 590, by = 10)
    ) +
  labs(title = "Mean Reaction Times (Self First)",
       x = "Mean reaction time (ms)",
       y = "Condition"
       ) +
  theme_grey(base_size = 12)
```



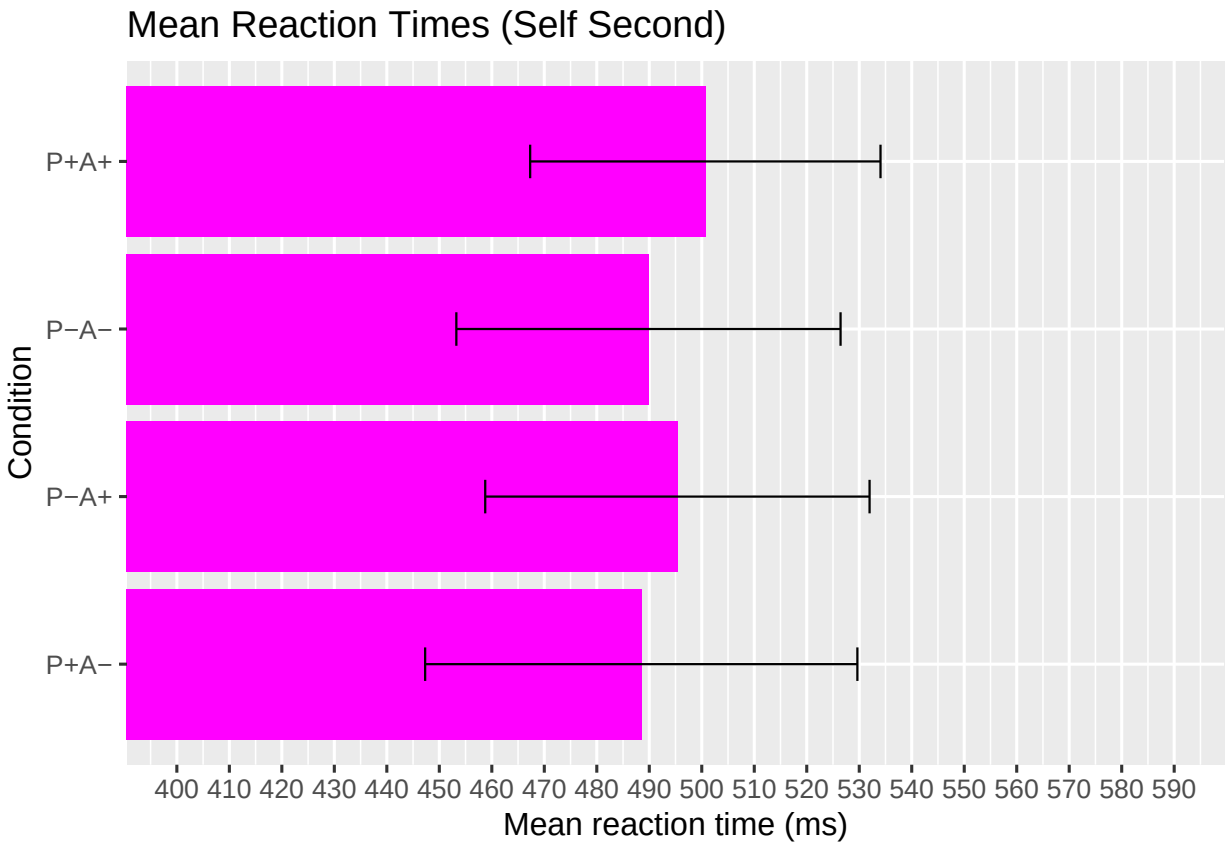

```
# plot mean RTs for smurf in second block
ggplot(means_self_first_smurf, aes(x = meanRT_smurf, y = Condition)) +
  geom_col(fill = "#0096FF") +
  geom_errorbarh(aes(xmin = meanRT_smurf - SE, xmax = meanRT_smurf + SE),
                 height = 0.2, size = 0.4) +
  coord_cartesian(xlim = c(400, 590)) +
  scale_x_continuous(
    breaks = seq(400, 590, by = 10)
    ) +
  labs(title = "Mean Reaction Times (Smurf Second)",
       x = "Mean reaction time (ms)",
       y = "Condition"
       ) +
  theme_grey(base_size = 12)
```




```
# plot mean RTs for smurf in first block
ggplot(means_smurf_first_smurf, aes(x = meanRT_smurf, y = Condition)) +
  geom_col(fill = "#0096FF") +
  geom_errorbarh(aes(xmin = meanRT_smurf - SE, xmax = meanRT_smurf + SE),
                 height = 0.2, size = 0.4) +
  coord_cartesian(xlim = c(400, 590)) +
  scale_x_continuous(
    breaks = seq(400, 590, by = 10)
    ) +
  labs(title = "Mean Reaction Times (Smurf First)",
       x = "Mean reaction time (ms)",
       y = "Condition"
       ) +
  theme_grey(base_size = 12)
```




```
# plot mean RTs for self in second block
ggplot(means_smurf_first_self, aes(x = meanRT_self, y = Condition)) +
  geom_col(fill = "#FF00FF") +
  geom_errorbarh(aes(xmin = meanRT_self - SE, xmax = meanRT_self + SE),
                 height = 0.2, size = 0.4) +
  coord_cartesian(xlim = c(400, 590)) +
  scale_x_continuous(
    breaks = seq(400, 590, by = 10)
    ) +
  labs(title = "Mean Reaction Times (Self Second)",
       x = "Mean reaction time (ms)",
       y = "Condition"
       ) +
  theme_grey(base_size = 12)
```



Compute means

```
meanRT_smurf <- mean(means_smurf$meanRT_smurf)

meanRT_self <- mean(means_self$meanRT_self)

meanRT_overall <- mean(means_overall$meanRT_overall)

mean_diff_smurf <- mean(df_smurf_wide$mean_diff)

mean_diff_self <- mean(df_self_wide$mean_diff)

mean_diff_self_first <- mean(df_self_first_self$mean_diff)

# count self first and smurf first participants
df_combined %>%
  distinct(subject_id, Agent_First) %>%
  count(Agent_First)
```

```
## # A tibble: 2 x 2
##   Agent_First     n
##   <chr>       <int>
## 1 Self           16
## 2 Smurf          15
```

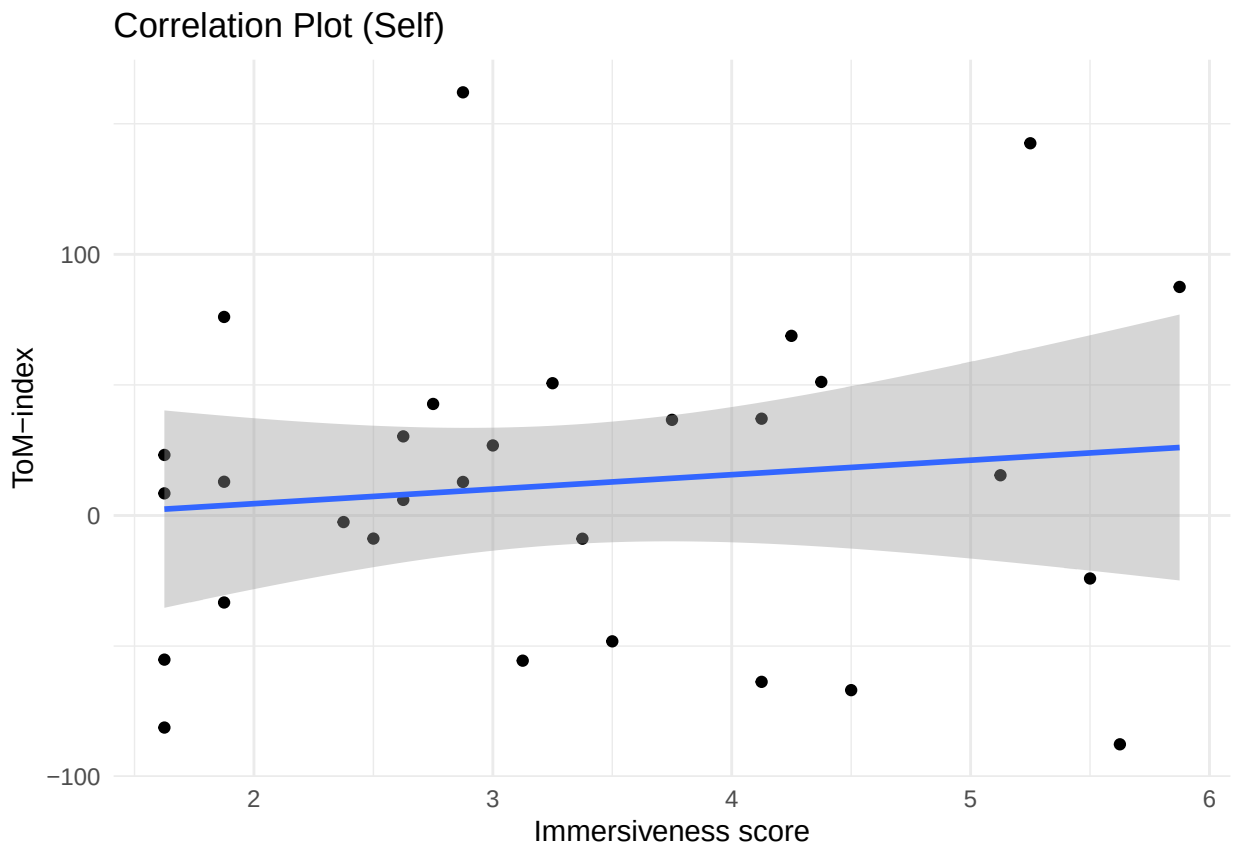
Questionnaire correlations

```
# filter out missing data
df_self_wide_filt <- df_self_wide %>%
  filter(subject_id != 1738)

df_smurf_wide_filt <- df_self_wide %>%
  filter(subject_id != 1738)

# correlation test self
correlationBF(df_self_wide_filt$score_self, df_self_wide_filt$mean_diff)
```

```
## Bayes factor analysis
## --------------
## [1] Alt., r=0.333 : 0.4738645 ±0%
##
## Against denominator:
##   Null, rho = 0
## ---
## Bayes factor type: BFcorrelation, Jeffreys-beta*
```

```
ggplot(df_self_wide_filt, aes(x = score_self, y = mean_diff)) +
  geom_point() +
  geom_smooth(method = "lm", se = TRUE) +
  theme_minimal() +
  labs(x = "Immersiveness score", y = "ToM-index",
       title = "Correlation Plot (Self)")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

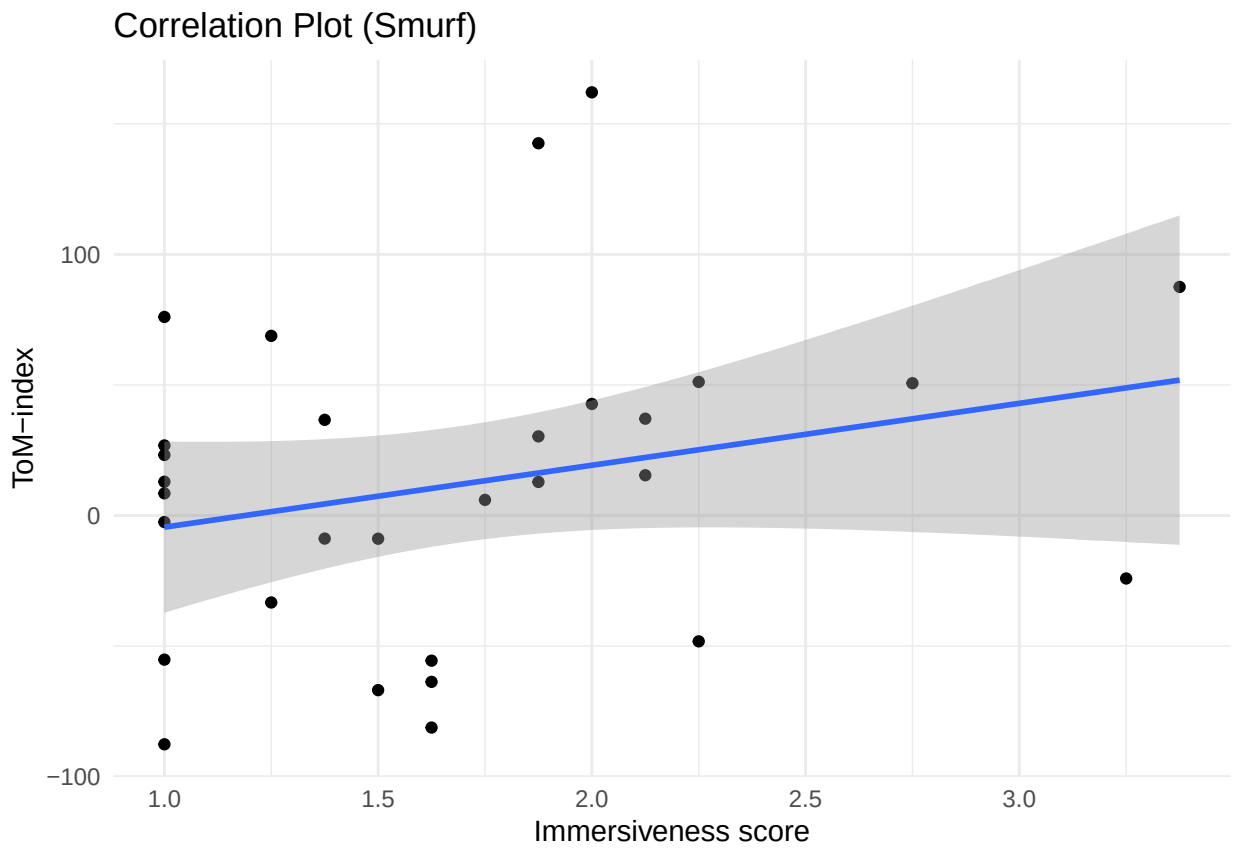



```
# correlation test smurf
correlationBF(df_smurf_wide_filt$score_smurf, df_smurf_wide_filt$mean_diff)
```

```
## Bayes factor analysis
## --------------
## [1] Alt., r=0.333 : 0.8699936 ±0%
##
## Against denominator:
##   Null, rho = 0
## ---
## Bayes factor type: BFcorrelation, Jeffreys-beta*
```

```
ggplot(df_smurf_wide_filt, aes(x = score_smurf, y = mean_diff)) +
  geom_point() +
  geom_smooth(method = "lm", se = TRUE) +
  theme_minimal() +
  labs(x = "Immersiveness score", y = "ToM-index",
       title = "Correlation Plot (Smurf)")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

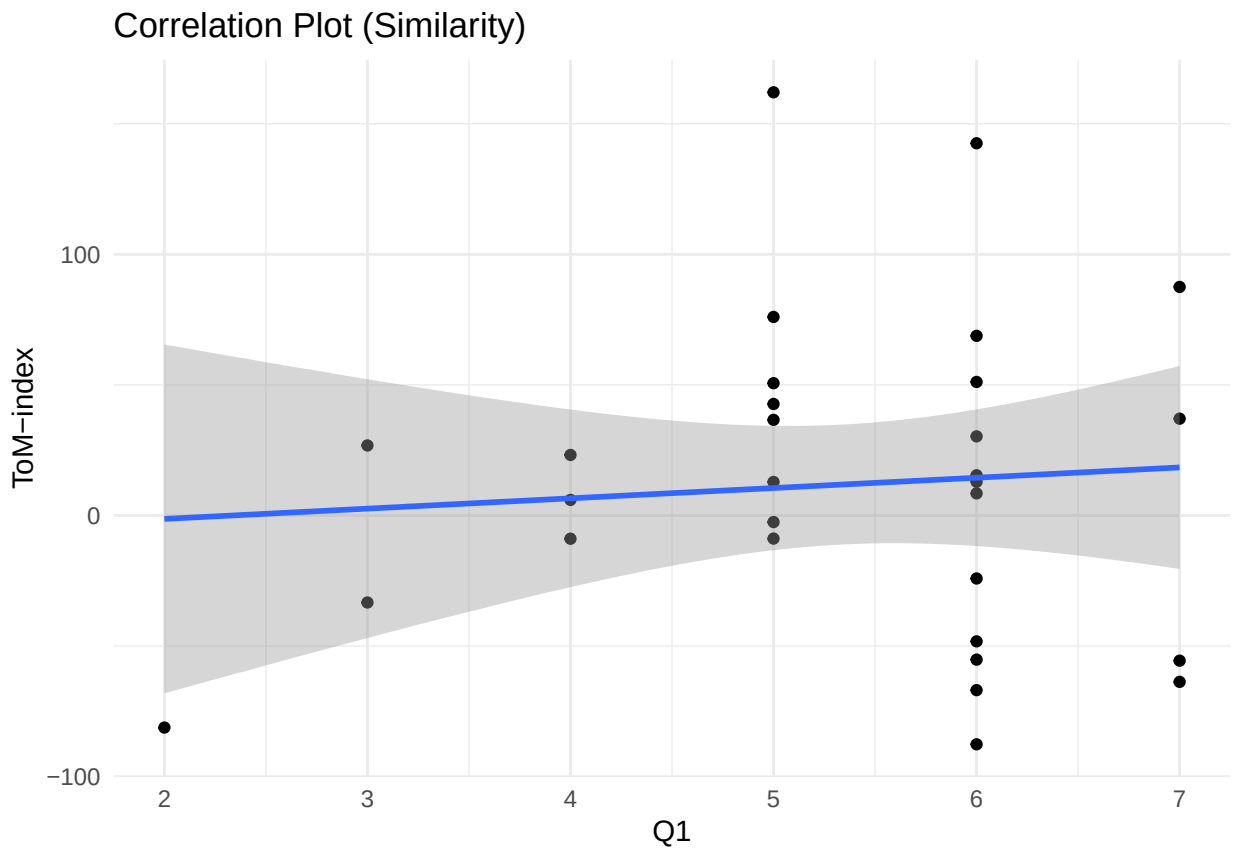



```
# correlation between similarity (Q1) and ToM-index in self trials
correlationBF(df_self_wide_filt$Q1_self, df_self_wide_filt$mean_diff)
```

```
## Bayes factor analysis
## --------------
## [1] Alt., r=0.333 : 0.431921 ±0%
##
## Against denominator:
##   Null, rho = 0
## ---
## Bayes factor type: BFcorrelation, Jeffreys-beta*
```

```
ggplot(df_self_wide_filt, aes(x = Q1_self, y = mean_diff)) +
  geom_point() +
  geom_smooth(method = "lm", se = TRUE) +
  theme_minimal() +
  labs(x = "Q1", y = "ToM-index",
       title = "Correlation Plot (Similarity)")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

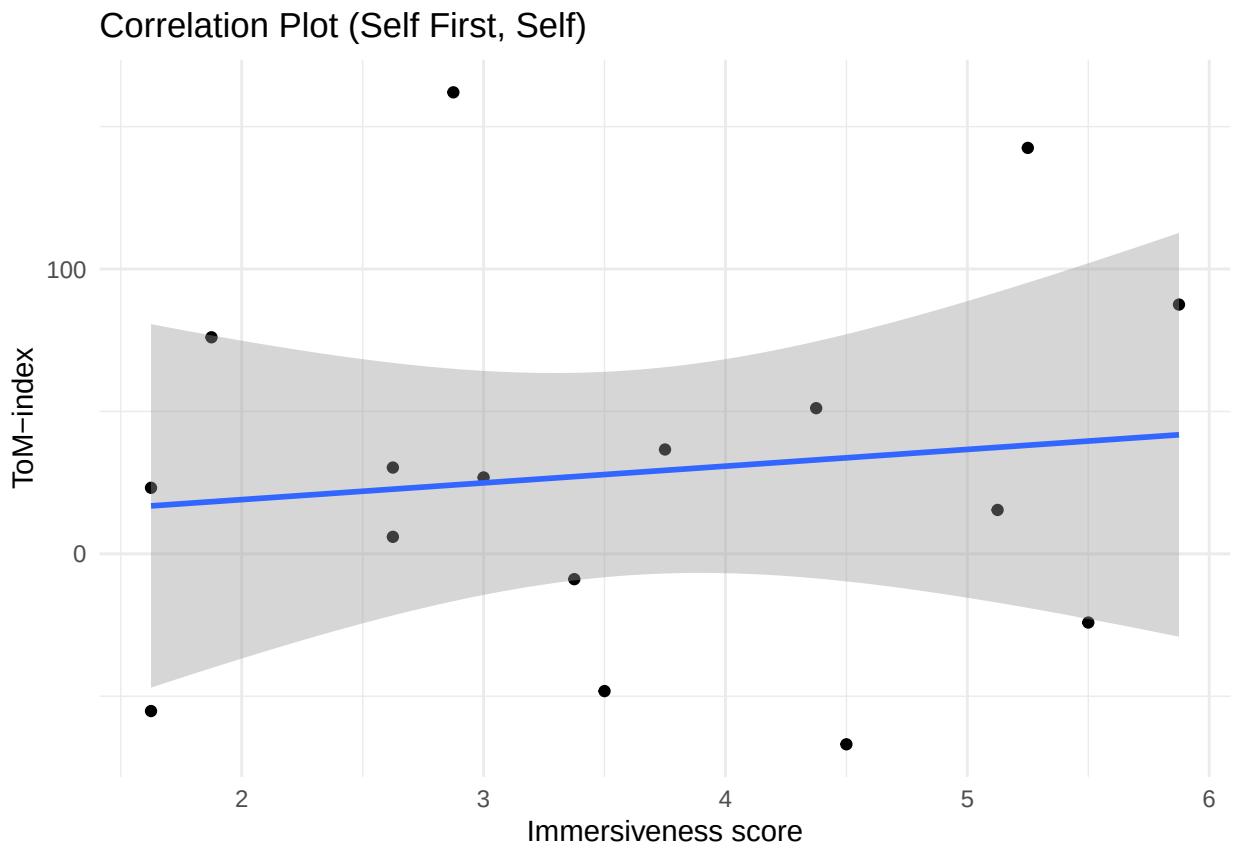



```
# correlation test self first self
df_self_first_self_filt <- df_self_first_self %>%
  filter(subject_id != 1738)

correlationBF(df_self_first_self_filt$score_self, df_self_first_self_filt$mean_diff)
```

```
## Bayes factor analysis
## --------------
## [1] Alt., r=0.333 : 0.5622684 ±0%
##
## Against denominator:
##   Null, rho = 0
## ---
## Bayes factor type: BFcorrelation, Jeffreys-beta*
```

```
ggplot(df_self_first_self_filt, aes(x = score_self, y = mean_diff)) +
  geom_point() +
  geom_smooth(method = "lm", se = TRUE) +
  theme_minimal() +
  labs(x = "Immersiveness score", y = "ToM-index",
       title = "Correlation Plot (Self First, Self)")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Questionnaire means

```
# filter out missing data
quest_scores_filt <- quest_scores %>%
  filter(subject_id != 1738)

# calculate mean scores
mean(quest_scores_filt$score_self)
```

```
## [1] 3.378788
```

```
mean(quest_scores_filt$score_smurf)
```

```
## [1] 1.636364
```

```
ttestBF(quest_scores_filt$score_self, quest_scores_filt$score_smurf, paired = TRUE)
```

```
## Bayes factor analysis
## --------------
## [1] Alt., r=0.707 : 4031618 ±0%
##
## Against denominator:
##   Null, mu = 0
## ---
## Bayes factor type: BFoneSample, JZS
```

```r
# mean Q8 score
mean(quest_scores_filt$Q8_self)
```

```
## [1] 3.181818
```

```r
mean(quest_scores_filt$Q8_smurf)
```

```
## [1] 2.393939
```

```r
# mean Q1 score
mean(quest_scores_filt$Q1_smurf)
```

```
## [1] 1.151515
```

```r
mean(quest_scores_filt$Q1_self)
```

```
## [1] 5.30303
```

```r
# mean perspective taking scores
(mean(quest_scores_filt$Q8_self) + mean(quest_scores_filt$Q7_self) + mean(quest_scores_filt$Q6_self)) /
```

```
## [1] 3.060606
```

```r
(mean(quest_scores_filt$Q8_smurf) + mean(quest_scores_filt$Q7_smurf) + mean(quest_scores_filt$Q6_smurf)
```

```
## [1] 2.282828
```

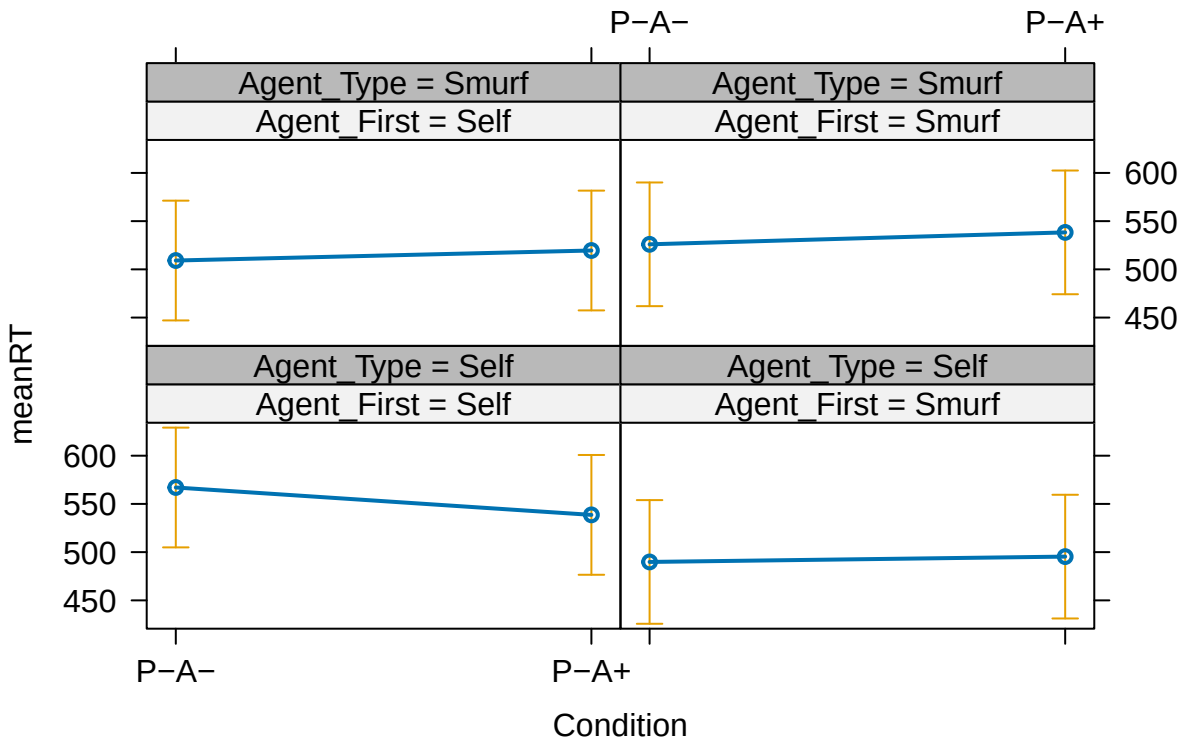
Linear mixed model

```r
# all factors
model <- lmer(
  meanRT ~ Condition * Agent_First * Agent_Type + (1 | subject_id),
  data = df_combined %>% filter(Condition %in% c('P-A-','P-A+'))
  )

summary(model)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: meanRT ~ Condition * Agent_First * Agent_Type + (1 | subject_id)
##    Data: df_combined %>% filter(Condition %in% c("P-A-", "P-A+"))
##
## REML criterion at convergence: 1339.6
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.04127 -0.52681 -0.03938  0.47150  2.58387
##
## Random effects:
##  Groups     Name        Variance Std.Dev.
##  subject_id (Intercept) 13482    116.11
##  Residual                2246     47.39
## Number of obs: 124, groups:  subject_id, 31
##
## Fixed effects:
##                                          Estimate Std. Error     df
## (Intercept)                                567.03      31.35  36.20
## ConditionP-A+                              -28.39      16.75  87.00
```

```
## Agent_FirstSmurf                                    -77.19      45.07  36.20
## Agent_TypeSmurf                                     -57.88      16.75  87.00
## ConditionP-A+:Agent_FirstSmurf                       33.91      24.09  87.00
## ConditionP-A+:Agent_TypeSmurf                        38.80      23.69  87.00
## Agent_FirstSmurf:Agent_TypeSmurf                     94.01      24.09  87.00
## ConditionP-A+:Agent_FirstSmurf:Agent_TypeSmurf     -31.94      34.06  87.00
##                                                   t value Pr(>|t|)
## (Intercept)                                        18.086  < 2e-16 ***
## ConditionP-A+                                      -1.695 0.093733 .
## Agent_FirstSmurf                                   -1.713 0.095342 .
## Agent_TypeSmurf                                    -3.455 0.000853 ***
## ConditionP-A+:Agent_FirstSmurf                      1.408 0.162757
## ConditionP-A+:Agent_TypeSmurf                       1.637 0.105176
## Agent_FirstSmurf:Agent_TypeSmurf                    3.903 0.000187 ***
## ConditionP-A+:Agent_FirstSmurf:Agent_TypeSmurf     -0.938 0.351008
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) CnP-A+ Agn_FS Agn_TS CnP-A+:A_FS CP-A+:A_T A_FS:A
## ConditnP-A+ -0.267
## Agnt_FrstSm -0.696  0.186
## Agnt_TypSmr -0.267  0.500  0.186
## CnP-A+:A_FS  0.186 -0.696 -0.267 -0.348
## CnP-A+:A_TS  0.189 -0.707 -0.131 -0.707  0.492
## Agn_FS:A_TS  0.186 -0.348 -0.267 -0.696  0.500       0.492
## CP-A+:A_FS: -0.131  0.492  0.189  0.492 -0.707      -0.696    -0.707
```

```
plot(allEffects(model))
```




```
emm_agent <- emmeans(model, ~ Agent_Type)
```

```
## NOTE: Results may be misleading due to involvement in interactions
```

```
emm_agent
```

```
##  Agent_Type emmean   SE   df lower.CL upper.CL
##  Self          523 21.7 31.4      478      567
##  Smurf         523 21.7 31.4      479      568
##
## Results are averaged over the levels of: Condition, Agent_First
## Degrees-of-freedom method: kenward-roger
## Confidence level used: 0.95
```

```
contrast(emm_agent, method = "pairwise")
```

```
##  contrast     estimate   SE df t.ratio p.value
##  Self - Smurf   -0.534 8.52 87  -0.063  0.9502
##
## Results are averaged over the levels of: Condition, Agent_First
## Degrees-of-freedom method: kenward-roger
```

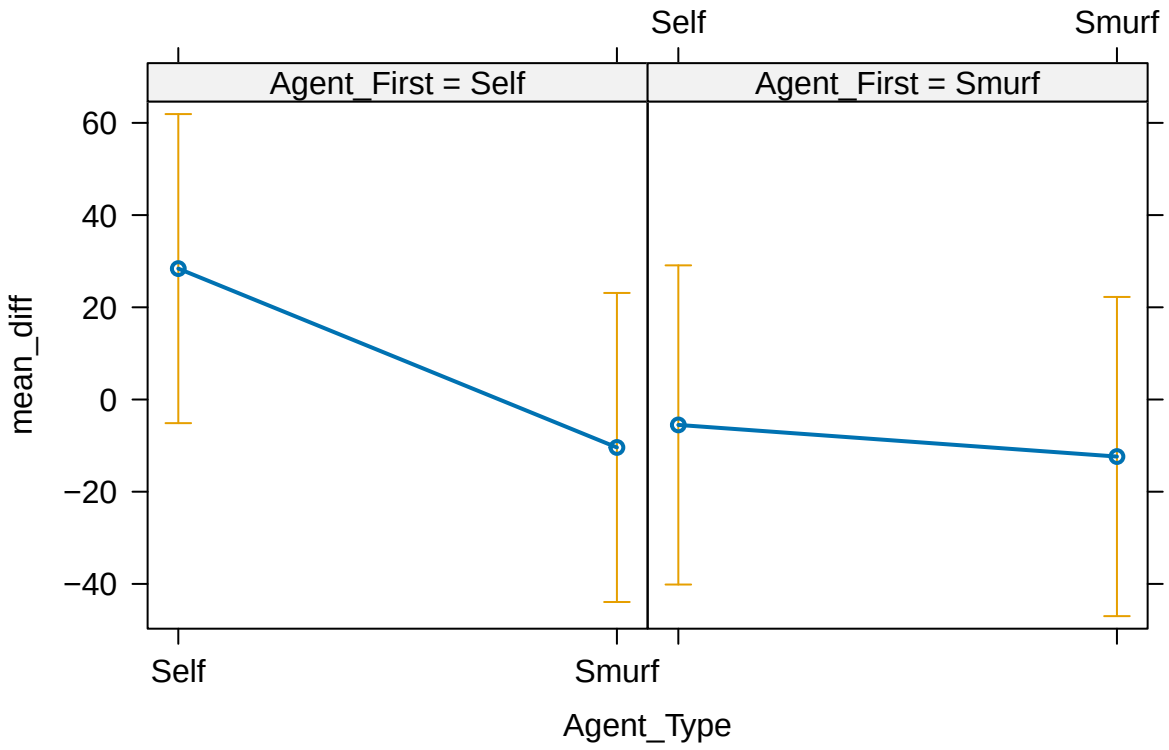
```
model2 <- lmer(
  mean_diff ~ Agent_Type * Agent_First + (1 | subject_id),
  data = df_wide
  )
```

```
## boundary (singular) fit: see help('isSingular')
```

```r
summary(model2)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: mean_diff ~ Agent_Type * Agent_First + (1 | subject_id)
##    Data: df_wide
##
## REML criterion at convergence: 663.3
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -1.67528 -0.79091  0.03663  0.51344  2.80264
##
## Random effects:
##  Groups     Name        Variance Std.Dev.
##  subject_id (Intercept)    0      0.00
##  Residual                4486     66.98
## Number of obs: 62, groups:  subject_id, 31
##
## Fixed effects:
##                                 Estimate Std. Error     df t value Pr(>|t|)
## (Intercept)                        28.39      16.74  58.00   1.696   0.0953 .
## Agent_TypeSmurf                   -38.80      23.68  58.00  -1.638   0.1068
## Agent_FirstSmurf                  -33.91      24.07  58.00  -1.409   0.1643
## Agent_TypeSmurf:Agent_FirstSmurf   31.94      34.04  58.00   0.938   0.3520
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) Agn_TS Agn_FS
## Agnt_TypSmr -0.707
## Agnt_FrstSm -0.696  0.492
## Agn_TS:A_FS  0.492 -0.696 -0.707
## optimizer (nloptwrap) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
```

```r
plot(allEffects(model2))
```

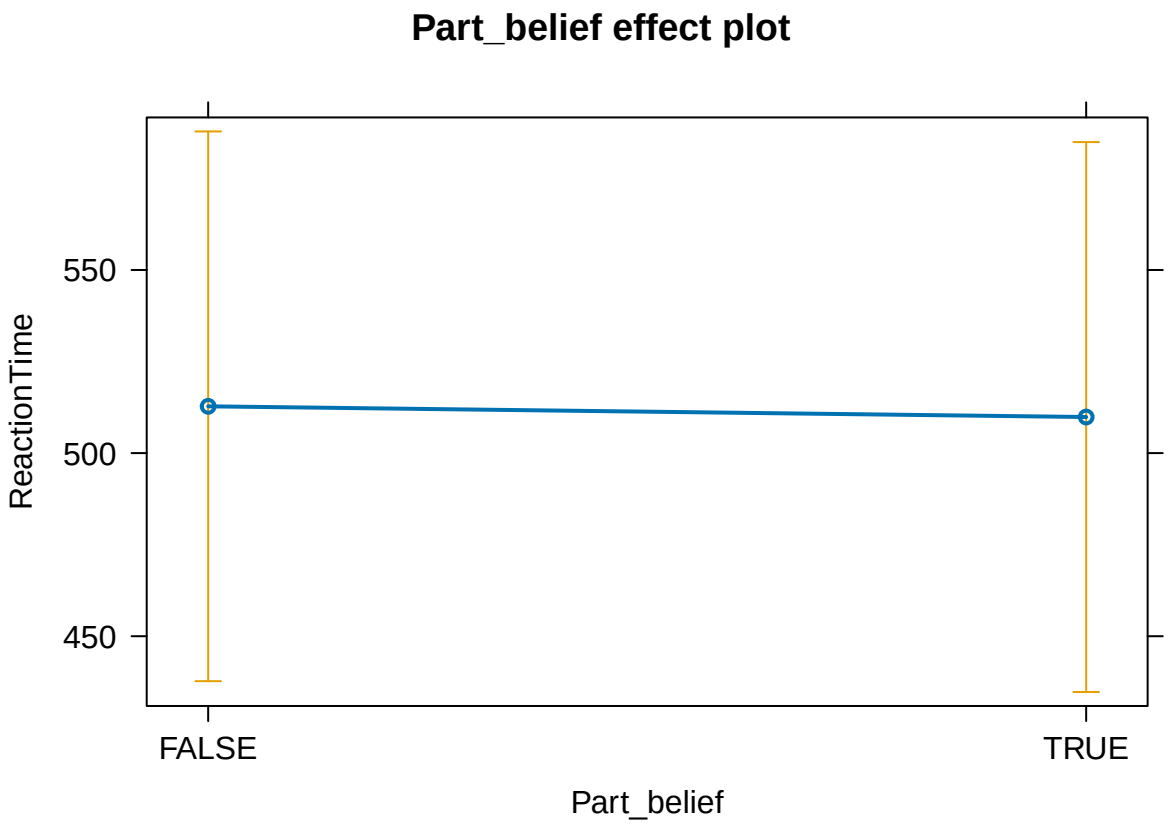



```
# participant belief
model3 <- lmer(
  ReactionTime ~ Part_belief + (1 | subject_id),
  data = df_final %>% filter(Agent_First == "Smurf")
  )

summary(model3)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: ReactionTime ~ Part_belief + (1 | subject_id)
##    Data: df_final %>% filter(Agent_First == "Smurf")
##
## REML criterion at convergence: 6791.9
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.3698 -0.6231 -0.1019  0.4554  3.9504
##
## Random effects:
##  Groups     Name        Variance Std.Dev.
##  subject_id (Intercept) 21494    146.61
##  Residual                7818     88.42
## Number of obs: 571, groups:  subject_id, 15
##
## Fixed effects:
```

```
##                 Estimate Std. Error      df t value Pr(>|t|)
## (Intercept)      512.774     38.209  14.246  13.420 1.77e-09 ***
## Part_beliefTRUE   -2.922      7.405 555.001  -0.395    0.693
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr)
## Prt_blfTRUE -0.095
```

```
plot(allEffects(model3))
```




```
# agent belief
model4 <- lmer(
  ReactionTime ~ Agent_belief + (1 | subject_id),
  data = df_final # %>% filter(Agent_First == "Self", Agent_Type == "Self")
  )

summary(model4)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: ReactionTime ~ Agent_belief + (1 | subject_id)
##    Data: df_final
##
## REML criterion at convergence: 14280.3
##
```

```
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.4430 -0.6243 -0.1409  0.4231  4.0797
##
## Random effects:
##  Groups     Name        Variance Std.Dev.
##  subject_id (Intercept) 11962    109.37
##  Residual                8524     92.32
## Number of obs: 1192, groups:  subject_id, 31
##
## Fixed effects:
##                  Estimate Std. Error       df t value Pr(>|t|)
## (Intercept)       517.320     20.006   31.059  25.858   <2e-16 ***
## Agent_beliefTRUE   -3.178      5.351 1159.993  -0.594    0.553
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr)
## Agnt_blTRUE -0.134
```

```
plot(allEffects(model4))
```