Temporal_binding

Dóra Jázmin Szájer

2025-12-11

#Data preprocessing

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(tidyr)
library(zoo)
```

```
## Warning: package 'zoo' was built under R version 4.4.3

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
data <- read.csv("C:/Users/szjgd/Documents/Cognitive Science Bsc/Perception and action/Exam project/expe

#Isolating mouse-tracking rows
mouse_tracking <- data %>%
  select(
    participant = mouseTracking.participant_id,
    trial = mouseTracking.trial_number,
    timestamp = mouseTracking.timestamp,
    x = mouseTracking.x,
    y = mouseTracking.y
  ) %>%
  drop_na(timestamp) #ensure timestamp exists
```

```
underage_participants <- data[data$participants.age < 18, ] #accessing participant IDs under 18
View(underage_participants)
mouse_tracking <- mouse_tracking[!mouse_tracking$participant %in% c(14, 15, 16, 17,
                                                                    20, 42, 44, 46, 52), ] #remove firs



#isolating questionnaire data
questionnaire <- data[ , 1:25]
questionnaire <- questionnaire[-(1:4), ] #remove first 4 rows due to testing
questionnaire <- questionnaire[questionnaire$participants.age >= 18 & !is.na(questionnaire$participants
rownames(questionnaire) <- NULL #reset rownames to sequential numbering
questionnaire <- na.omit(questionnaire)


#isolating correctness
correctness <- data[ , 26:33]
correctness <- correctness[!correctness$responses.participant_id %in% c(14, 15, 16, 17,
                                                                    20, 42, 44, 46, 52), ]

gender_counts_full <- table(data$participants.gender)
gender_counts_full
```

```
##
##        female   male  other
##  32938     28     31      3
```

#Dividing participants based on their ADHD score

```
inattention_scores <- questionnaire %>%
  select(
    participant = participants.id,
    q1_3p = questionnaires.q1,
    q2_3p = questionnaires.q2,
    q3_3p = questionnaires.q3,
    q9_3p = questionnaires.q9,
    q12_3p = questionnaires.q12,
    q4 = questionnaires.q4,
    q7 = questionnaires.q7,
    q8 = questionnaires.q8,
    q10 = questionnaires.q10,
    q11 = questionnaires.q11
  )

#first 5 rows have to be a score above 2 & second 5 rows above 3
inattention_group <- inattention_scores[
  rowSums(inattention_scores[, 2:6] > 2) == 5 &
  rowSums(inattention_scores[, 7:11] > 3) == 5,
]
non_inattention_group <- inattention_scores[
  rowSums(inattention_scores[, 2:6] <= 2) == 5 &
  rowSums(inattention_scores[, 7:11] <= 3) == 5,
]
some_inattention_group <- inattention_scores[!inattention_scores$participant %in% c(31, 32, 48, 65, 25,

#averaging scores for independent variable
inattention_scores$total_score <- rowSums(inattention_scores[, 2:11])


#first 2 rows have to be a score above 2 & second 6 rows above 3
hyperactivity_scores <- questionnaire %>%
  select(
    participant = participants.id,
    q16_3p = questionnaires.q16,
    q18_3p = questionnaires.q18,
    q5 = questionnaires.q5,
    q6 = questionnaires.q6,
    q13 = questionnaires.q13,
    q14 = questionnaires.q14,
    q15 = questionnaires.q15,
    q17 = questionnaires.q17,
  )
hyperactivity_group <- hyperactivity_scores[
  rowSums(hyperactivity_scores[, 2:3] > 2) == 2 &
  rowSums(hyperactivity_scores[, 4:9] > 3) == 6,
]
non_hyperactivity_group <- hyperactivity_scores[
  rowSums(hyperactivity_scores[, 2:3] <= 2) == 2 &
  rowSums(hyperactivity_scores[, 4:9] <= 3) == 6,
]
some_hyperactivity_group <- hyperactivity_scores[!hyperactivity_scores$participant %in% c(19, 38, 48, 6

combined_group_IDs <- intersect(inattention_group$participant, hyperactivity_group$participant)
combined_group <- inattention_group %>% filter(participant %in% combined_group_IDs)

#total scores for independent variable
hyperactivity_scores$total_score <- rowSums(hyperactivity_scores[, 2:9])

#adding total score to the main df of adhd scores
questionnaire$total_score <- rowSums(questionnaire[, 7:24])
```

#ADHD score divisions for stats analysis

```
#median split of total adhd scores
med <- median(questionnaire$total_score, na.rm = TRUE)

#create ad hoc groups
questionnaire$adhd_group <- ifelse(questionnaire$total_score <= med,
                                   "low_adhd",
                                   "high_adhd")
group_distribution <- table(questionnaire$adhd_group)
group_distribution
```

```
##
## high_adhd  low_adhd
##        26        27
```

```
ranges <- tapply(questionnaire$total_score,     #check the actual median and ranges per group
       questionnaire$adhd_group,
       summary)
ranges
```

```
## $high_adhd
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   56.00   59.00   63.00   64.12   67.75   82.00 
## 
## $low_adhd
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   26.00   37.50   42.00   43.33   50.50   55.00
```

```
gender_counts <- table(questionnaire$adhd_group, questionnaire$participants.gender)
gender_counts
```

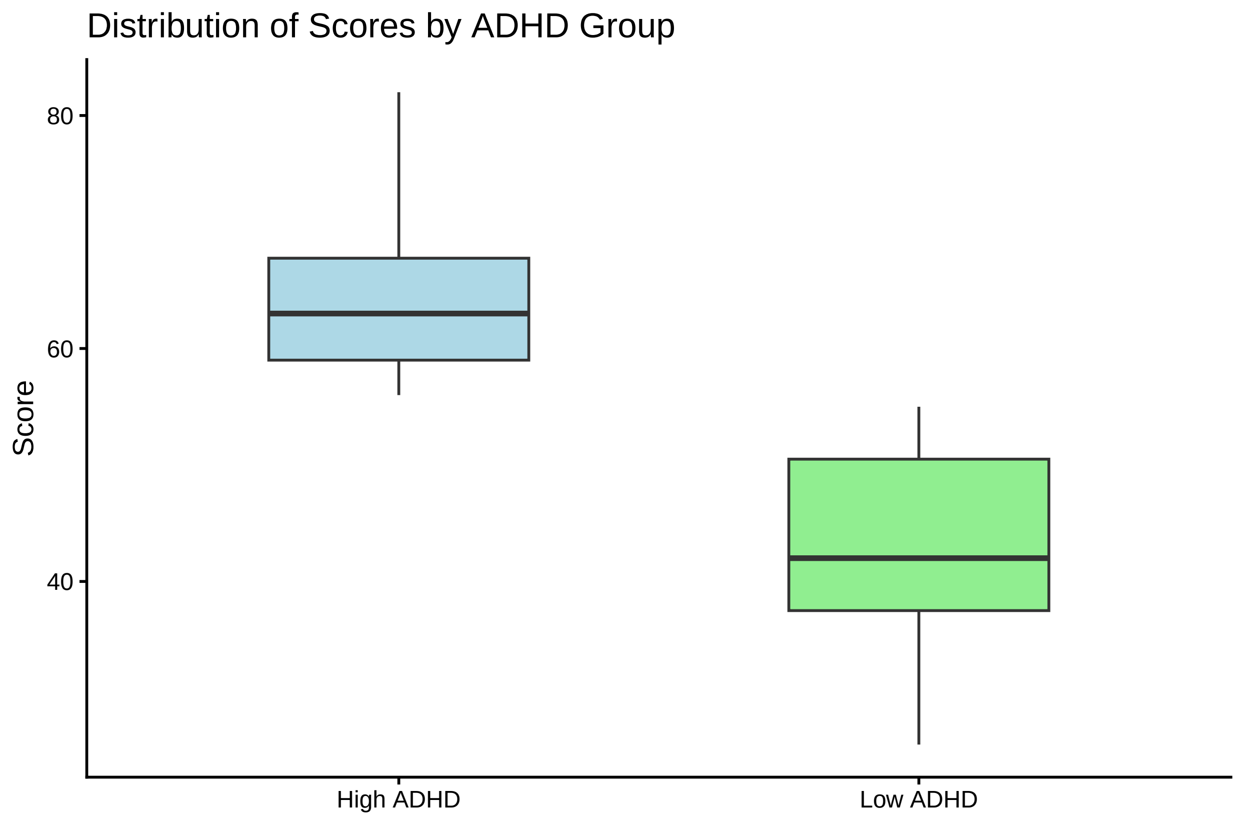
```
## 
##             female male other
##   high_adhd     15    9     2
##   low_adhd       8   18     1
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.4.3
```

```
# create dataframe from summaries
df_summary <- data.frame(
  group = c("High ADHD", "Low ADHD"),
  ymin  = c(56, 26),
  lower = c(59, 37.5),
  middle= c(63, 42),
  upper = c(67.75, 50.5),
  ymax  = c(82, 55)
)

ggplot(df_summary, aes(x = group)) +
  geom_boxplot(
    aes(
      ymin = ymin,
      lower = lower,
      middle = middle,
      upper = upper,
      ymax = ymax,
      fill = group
    ),
    stat = "identity",
    width = 0.5
  ) +
  labs(
```

```r
    x = "",
    y = "Score",
    title = "Distribution of Scores by ADHD Group"
  ) +
  theme_classic() +
  scale_fill_manual(values = c("lightblue", "lightgreen")) +
  theme(legend.position = "none")
```



For the paper this can be reported as: Participants were divided into ad hoc low- and high-ADHD symptom groups using a median split on total questionnaire scores (Low: n = 27; High: n = 26).In the low-ADHD group, 15 participants identified as female, 9 as male and 2 as other, whereas in the high-ADHD group, 8 participants identified as female, 18 as male and 1 as other.

#Accessing reaction times

```r
rt_per_trial <- mouse_tracking %>%
  group_by(participant, trial) %>%
  summarize(
    RT = max(timestamp) - min(timestamp),
  )
```

```
## `summarise()` has grouped output by 'participant'. You can override using the
## `.groups` argument.
```

```
rt_per_trial
```

```
## # A tibble: 1,053 x 3
## # Groups:   participant [53]
##    participant trial    RT
##          <int> <int> <int>
##  1          18     1  4815
##  2          18     2  1212
##  3          18     3   567
##  4          18     4   421
##  5          18     5   488
##  6          18     6   286
##  7          18     7   251
##  8          18     8  2516
##  9          18     9   509
## 10          18    10   727
## # i 1,043 more rows
```

```
#filtering out participants who used their phone
phone_participants <- rt_per_trial %>%
  group_by(participant) %>%
  summarise(
    n_trials = n(),
    zero_rt_trials = sum(RT == 0, na.rm = TRUE),
    zero_rt_prop = zero_rt_trials / n_trials
  ) %>%
  filter(zero_rt_prop >= 0.8)

#removing them from the data frames
rt_clean <- rt_per_trial %>%
  anti_join(phone_participants, by = "participant")
```

We will have two statistical models: - Accuracy model: logistic mixed-effects model - DV: correctness, IVs: delay, ADHD scores - Reaction time model: linear mixed-effects model - DV: reaction time, IVs: delay, ADHD scores

Prep for models

```
#renaming column names in dfs to join columns for the models
questionnaire <- questionnaire %>%
  rename(
    participant = participants.id
  )

inattention_scores <- inattention_scores %>%
  rename(
    in_score = total_score
  )

hyperactivity_scores <- hyperactivity_scores %>%
```

```r
  rename(
    hp_score = total_score
  )

#adding hyperactivity scores and inattention scores
questionnaire <- questionnaire %>%
  left_join(
    inattention_scores %>%
      select(participant, last_col()),
    by = "participant"
  )

questionnaire <- questionnaire %>%
  left_join(
    hyperactivity_scores %>%
      select(participant, last_col()),
    by = "participant"
  )

#Joining all needed information into one df
correctness <- correctness %>%
  rename(
    participant = responses.participant_id
  )

df_model <- correctness %>%
  left_join(questionnaire, by = "participant")

#changing classes
df_model$participant <- as.factor(df_model$participant)
```

Next, we have to check for: - Independence: handled via random effects - Multicollinearity - Correct random effects - Influential observations - Overdispersion

#Accuracy model - logistic mixed effects regression

```r
library(lme4)
```

```
## Warning: package 'lme4' was built under R version 4.4.3
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```r
model_accuracy <- glmer(
  responses.correct ~ responses.delay_ms + in_score + hp_score +
    (1 | participant),
  family = binomial(link = "logit"),
  data = df_model,
  control = glmerControl(optimizer = "bobyqa")
)
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, : Model is nearly uniden
##  - Rescale variables?
```

```
#checking for convergence & singularity
summary(model_accuracy)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: responses.correct ~ responses.delay_ms + in_score + hp_score +
##     (1 | participant)
##    Data: df_model
## Control: glmerControl(optimizer = "bobyqa")
##
##       AIC       BIC    logLik -2*log(L)  df.resid
##    1212.7    1237.6    -601.4    1202.7      1055
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -5.0413 -0.6337 -0.3285  0.6764  3.3526
##
## Random effects:
##  Groups      Name        Variance Std.Dev.
##  participant (Intercept) 0.7618   0.8728
## Number of obs: 1060, groups:  participant, 53
##
## Fixed effects:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -1.9958383  0.6232671  -3.202  0.00136 **
## responses.delay_ms  0.0118854  0.0009163  12.971  < 2e-16 ***
## in_score            0.0286169  0.0223542   1.280  0.20049
## hp_score           -0.0226947  0.0292198  -0.777  0.43734
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) rspn._ in_scr
## rspnss.dly_ -0.200
## in_score    -0.417  0.036
## hp_score    -0.422 -0.023 -0.615
## optimizer (bobyqa) convergence code: 0 (OK)
## Model is nearly unidentifiable: very large eigenvalue
##  - Rescale variables?
```

```
#checking for multicollinearity
library(performance)
```

```
## Warning: package 'performance' was built under R version 4.4.3
```

```
check_collinearity(model_accuracy)
```

```
## # Check for Multicollinearity
##
## Low Correlation
##
##                Term  VIF       VIF 95% CI adj. VIF Tolerance Tolerance 95% CI
##  responses.delay_ms 1.00 [1.00,      Inf]     1.00      1.00     [0.00, 1.00]
##            in_score 1.61 [1.49,     1.76]     1.27      0.62     [0.57, 0.67]
##            hp_score 1.61 [1.49,     1.76]     1.27      0.62     [0.57, 0.67]
```

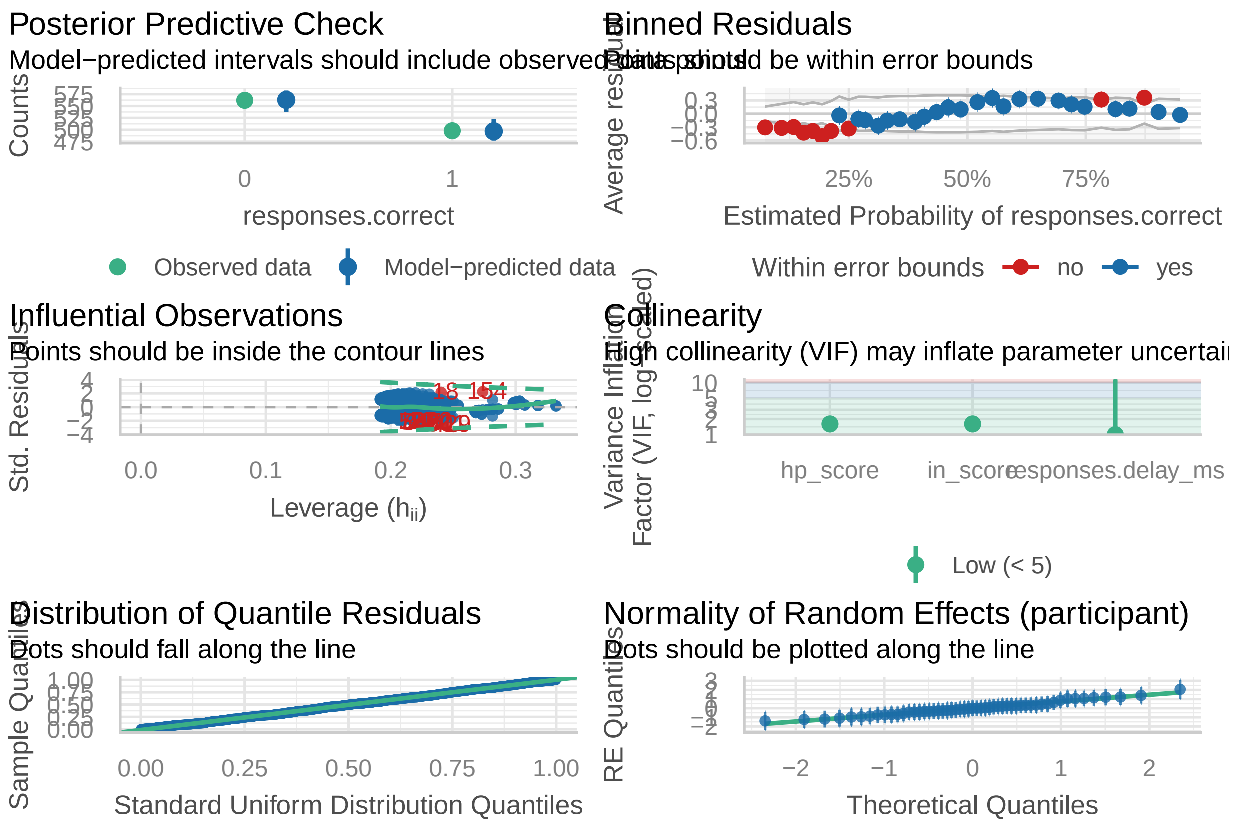
```
#checking for overdispersion
check_overdispersion(model_accuracy)
```

```
## # Overdispersion test
##
##  dispersion ratio = 0.998
##           p-value = 0.976

## No overdispersion detected.
```

```
#Random effects justification (checking for variance)
VarCorr(model_accuracy)
```

```
##  Groups      Name        Std.Dev.
##  participant (Intercept) 0.87282
```

```
#Linearity of continuous predictors
check_model(model_accuracy)
```



```r
#we got very large eigenvalues: have to rescale predictors
df_model <- df_model |>
  mutate(
    delay_z = scale(responses.delay_ms),
    in_z = scale(in_score),
    hp_z = scale(hp_score)
  )

model_acc <- glmer(
  responses.correct ~ delay_z + in_z + hp_z + (1 | participant),
  data = df_model,
  family = binomial,
  control = glmerControl(optimizer = "bobyqa")
)

summary(model_acc)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: responses.correct ~ delay_z + in_z + hp_z + (1 | participant)
##    Data: df_model
## Control: glmerControl(optimizer = "bobyqa")
##
##       AIC       BIC    logLik -2*log(L)  df.resid
##    1212.7    1237.6    -601.4    1202.7      1055
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -5.0414 -0.6337 -0.3285  0.6764  3.3527
##
## Random effects:
##  Groups      Name        Variance Std.Dev.
##  participant (Intercept) 0.7618   0.8728
## Number of obs: 1060, groups:  participant, 53
##
## Fixed effects:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.12744    0.14034  -0.908    0.364
## delay_z      1.08555    0.08369  12.971   <2e-16 ***
## in_z         0.22916    0.17899   1.280    0.200
## hp_z        -0.13789    0.17757  -0.776    0.437
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##         (Intr) dely_z in_z
## delay_z  0.013
## in_z     0.003  0.036
## hp_z     0.001 -0.023 -0.615
```

Model diagnostics indicated acceptable fit. Posterior predictive checks showed that observed accuracy rates were well captured by the model. Residual diagnostics revealed minor deviations at low predicted probabilities, which are common in trial-level logistic mixed-effects models. No problematic collinearity or influential observations were detected.

A generalized linear mixed-effects model with a binomial link function was fitted to predict trial-level response accuracy. Delay, inattentive symptom scores, and hyperactive/impulsive symptom scores (all z-standardized) were included as fixed effects, with random intercepts for participants.

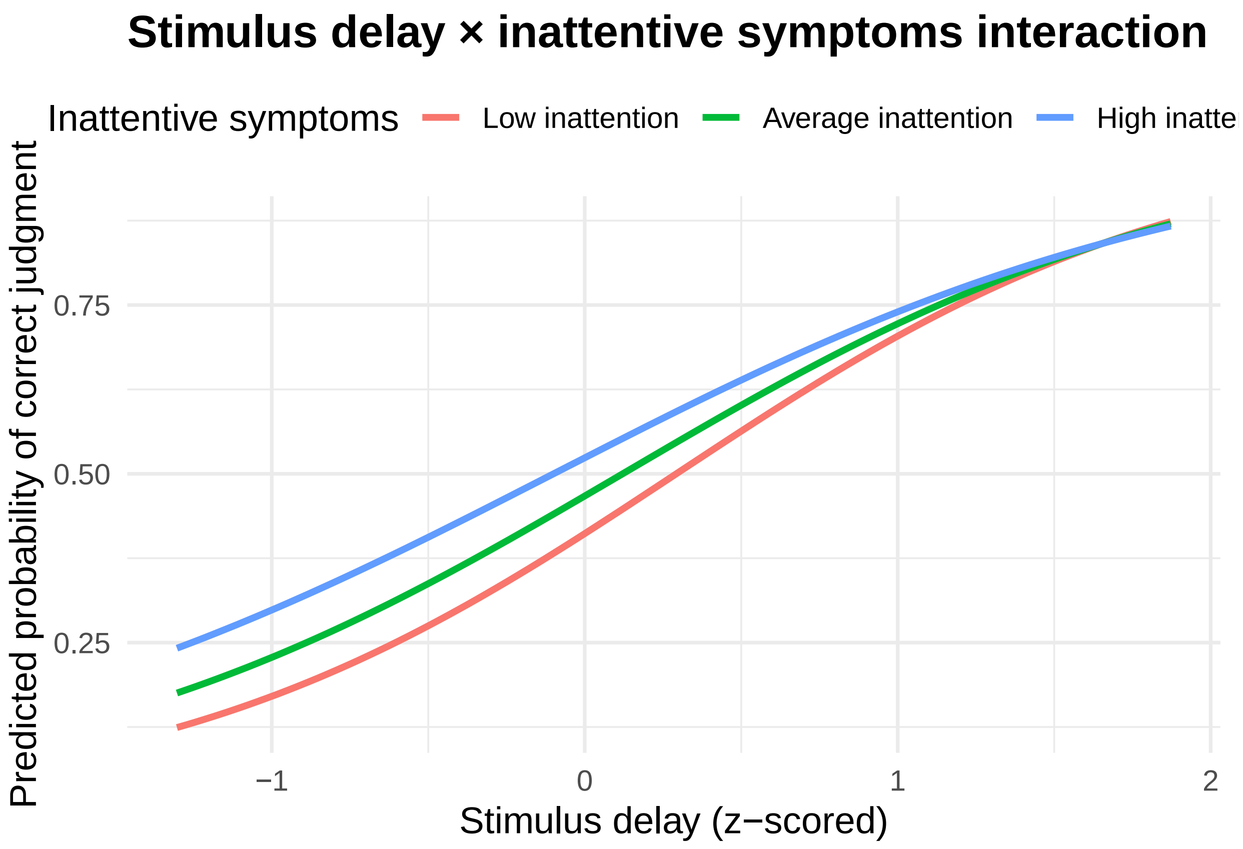
Accuracy was strongly predicted by delay, such that higher delays were associated with a higher probability of a correct response (= 1.09, SE = 0.08, z = 12.97, p < .001). Inattentive and hyperactive/impulsive symptom scores did not significantly predict accuracy (ps > .20). Substantial between-participant variability in baseline accuracy was observed (SD = 0.87).

Accuracy model - inattentiveness interaction

```r
# does the increase in accuracy as stimulus delay grows is steeper, flatter or shifted depending on ina
inattentive_interaction_model <- glmer(
  responses.correct ~ delay_z * in_z + hp_z + (1 | participant),
  data = df_model,
  family = binomial,
  control = glmerControl(optimizer = "bobyqa")
)
summary(inattentive_interaction_model)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: responses.correct ~ delay_z * in_z + hp_z + (1 | participant)
##    Data: df_model
## Control: glmerControl(optimizer = "bobyqa")
##
##       AIC       BIC    logLik -2*log(L)  df.resid
##    1212.0    1241.8    -600.0    1200.0      1054
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -5.0568 -0.6398 -0.3239  0.7012  3.7124
##
## Random effects:
##  Groups      Name        Variance Std.Dev.
##  participant (Intercept) 0.7517   0.867
## Number of obs: 1060, groups:  participant, 53
##
## Fixed effects:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.13150    0.13973  -0.941   0.3467
## delay_z       1.08740    0.08391  12.960   <2e-16 ***
## in_z          0.22646    0.17819   1.271   0.2038
## hp_z         -0.13197    0.17685  -0.746   0.4555
## delay_z:in_z -0.13693    0.08195  -1.671   0.0948 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) dely_z in_z   hp_z
## delay_z      0.011
## in_z        -0.006  0.032
## hp_z         0.000 -0.020 -0.615
## delay_z:n_z  0.014 -0.066  0.006 -0.016
```

```r
df_model <- na.omit(df_model)

library(ggplot2)

#create values for prediction for plotting
pred_df <- expand.grid(
  delay_z = seq(
    from = min(df_model$delay_z),
    to   = max(df_model$delay_z),
    length.out = 100
  ),
  in_z = c(-1, 0, 1),    # low, mean, high inattentive symptoms
  hp_z = 0              # hold hyperactivity at the mean
)

pred_df$predicted_prob <- predict(
  inattentive_interaction_model,
  newdata = pred_df,
  type = "response",
  re.form = NA     #ensuring population-level effects not individual participants
)

# Label inattentive levels
pred_df$in_label <- factor(
  pred_df$in_z,
  levels = c(-1, 0, 1),
  labels = c("Low inattention", "Average inattention", "High inattention")
)

ggplot(pred_df, aes(x = delay_z, y = predicted_prob, color = in_label)) +
  geom_line(linewidth = 1.2) +
  labs(
    x = "Stimulus delay (z-scored)",
    y = "Predicted probability of correct judgment",
    color = "Inattentive symptoms",
    title = "Stimulus delay × inattentive symptoms interaction"
  ) +
  theme_minimal(base_size = 14) +
  theme(
    legend.position = "top",
    plot.title = element_text(face = "bold")
  )
```



Accuracy increased robustly as the temporal delay between stimuli increased (= 1.09, SE = 0.08, p < .001). Inattentive and hyperactive/impulsive symptom scores did not predict overall accuracy. However, a marginal interaction between stimulus delay and inattentive symptoms was observed ($= -0.14$, SE = 0.08, p = .095), suggesting that the increase in accuracy with greater temporal separation was attenuated in participants with higher inattentive symptom levels.

Predicted probabilities derived from the mixed-effects model revealed that accuracy increased monotonically with stimulus delay across all participants. However, inattentive symptom severity modulated this relationship such that individuals with higher inattentive traits showed higher accuracy at shorter delays, with group differences diminishing as temporal separation increased. This pattern suggests differences in temporal integration rather than global performance deficits.

The results suggest the temporal binding theory of: Reduced temporal integration / noisier timing in adults with ADHD → earlier segregation → less tolerance for simultaneity (Because temporal integration is reduced, events that occur close together are segregated into separate perceptual units earlier than usual)

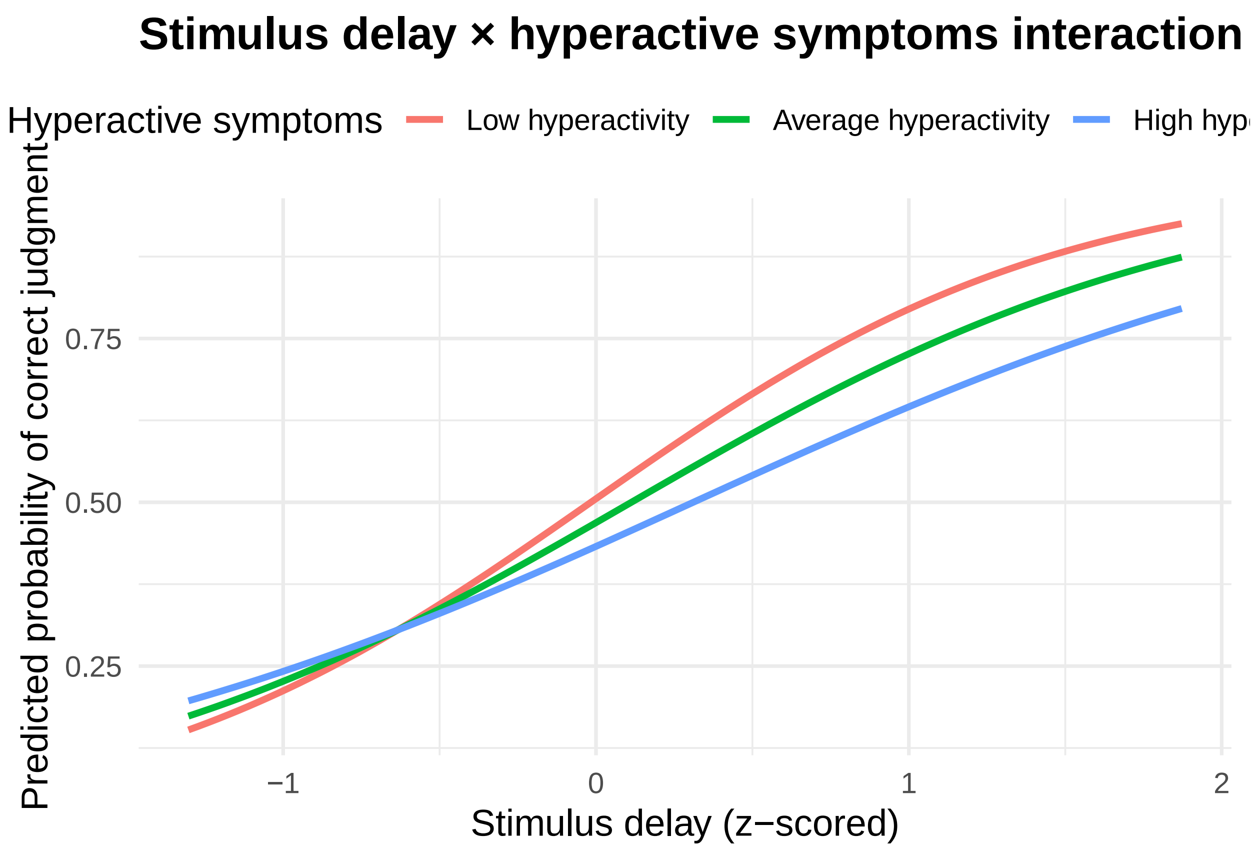
Accuracy model - hyperactivity interaction

```r
hyperactivity_interaction_model <- glmer(
  responses.correct ~ delay_z * hp_z +  in_z + (1 | participant),
  data = df_model,
  family = binomial,
  control = glmerControl(optimizer = "bobyqa")
)
summary(hyperactivity_interaction_model)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: responses.correct ~ delay_z * hp_z + in_z + (1 | participant)
##    Data: df_model
## Control: glmerControl(optimizer = "bobyqa")
##
##       AIC       BIC    logLik -2*log(L)  df.resid
##    1206.6    1236.4    -597.3    1194.6      1054
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.7853 -0.6376 -0.3232  0.7066  3.7187
##
## Random effects:
##  Groups      Name        Variance Std.Dev.
##  participant (Intercept) 0.7537   0.8681
## Number of obs: 1060, groups:  participant, 53
##
## Fixed effects:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.12455    0.14001  -0.890  0.37368
## delay_z       1.10209    0.08511  12.949  < 2e-16 ***
## hp_z         -0.14633    0.17694  -0.827  0.40822
## in_z          0.24086    0.17838   1.350  0.17694
## delay_z:hp_z -0.23094    0.08176  -2.825  0.00473 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Correlation of Fixed Effects:
##             (Intr) dely_z hp_z   in_z
## delay_z      0.017
## hp_z        -0.015 -0.035
## in_z         0.003  0.045 -0.613
## dely_z:hp_z -0.012 -0.156  0.023 -0.024
```

```
# Create values for prediction for plotting
pred_df <- expand.grid(
  delay_z = seq(
    from = min(df_model$delay_z),
    to   = max(df_model$delay_z),
    length.out = 100
  ),
  hp_z = c(-1, 0, 1),    # low, mean, high hyperactivity symptoms
  in_z = 0               # hold inattention at the mean
)

# Get predicted probabilities from the hyperactivity interaction model
pred_df$predicted_prob <- predict(
  hyperactivity_interaction_model,
  newdata = pred_df,
  type = "response",
  re.form = NA           # population-level effects only
)

# Label hyperactivity levels
pred_df$hp_label <- factor(
  pred_df$hp_z,
  levels = c(-1, 0, 1),
  labels = c("Low hyperactivity", "Average hyperactivity", "High hyperactivity")
)

# Plot
ggplot(pred_df, aes(x = delay_z, y = predicted_prob, color = hp_label)) +
  geom_line(linewidth = 1.2) +
  labs(
    x = "Stimulus delay (z-scored)",
    y = "Predicted probability of correct judgment",
    color = "Hyperactive symptoms",
    title = "Stimulus delay × hyperactive symptoms interaction"
  ) +
  theme_minimal(base_size = 14) +
  theme(
    legend.position = "top",
    plot.title = element_text(face = "bold")
  )
```



Predicted probabilities derived from the mixed-effects model revealed that accuracy increased monotonically with stimulus delay across all participants. However, hyperactive symptom severity modulated this relationship such that individuals with higher hyperactive traits showed lower accuracy at longer delays, with group differences diminishing at shorter delays. This pattern suggests differences in temporal integration rather than a global performance deficit.

These results are consistent with the temporal binding theory of:

Reduced temporal integration / noisier timing in adults with ADHD → earlier segregation of events → less tolerance for simultaneity

Specifically, because temporal integration is reduced, individuals with higher hyperactive traits appear to benefit less from longer temporal separations, perceiving events as segregated sooner than those with lower hyperactive traits.

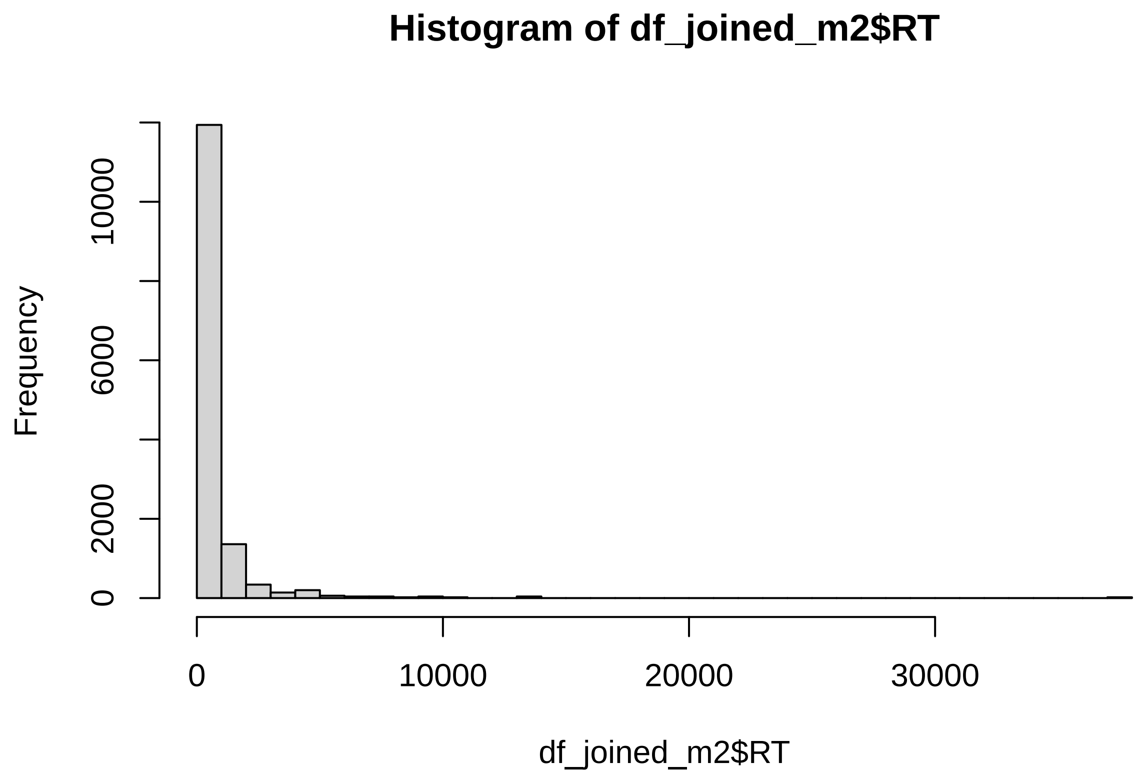
Reaction time model - linear mixed effects model

```r
#removing phone participants from df_model
valid_participants <- unique(rt_clean$participant)
df_model2 <- df_model %>%
  dplyr::filter(participant %in% valid_participants)

#join them
df_model2 <- df_model2 %>%
  dplyr::mutate(participant = as.character(participant))
```

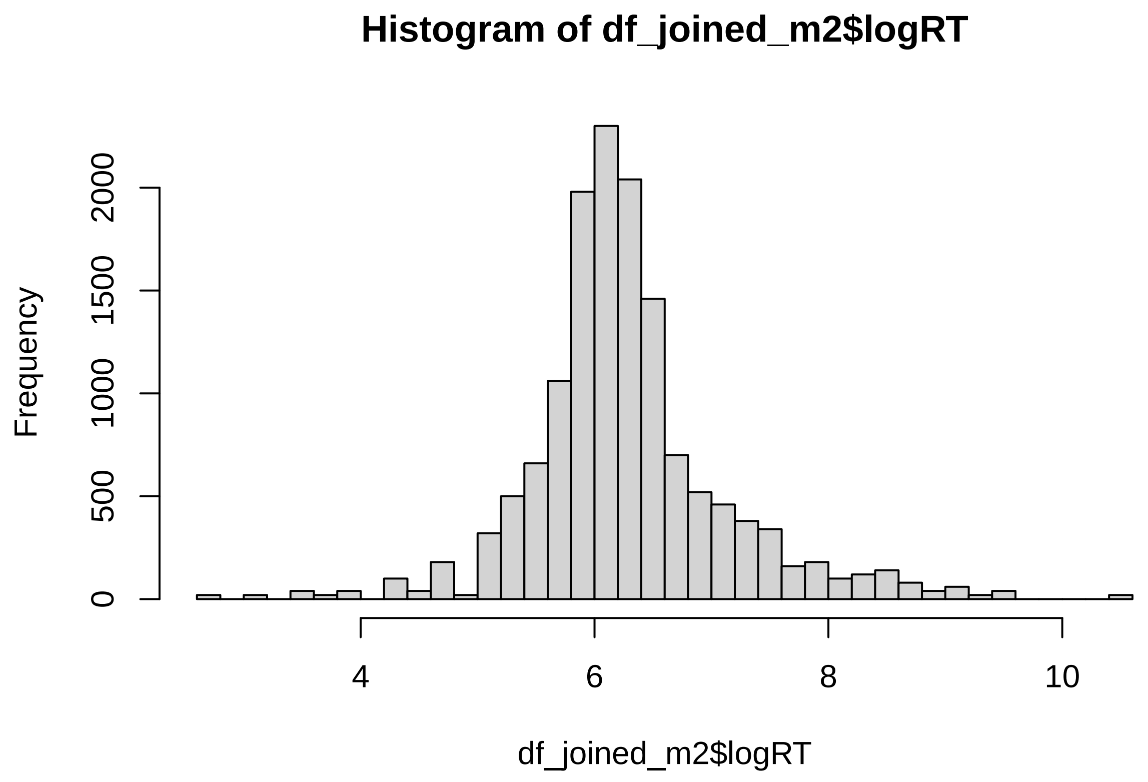
```r
rt_clean <- rt_clean %>%
  dplyr::mutate(participant = as.character(participant))

df_joined_m2 <- df_model2 %>%
  left_join(rt_clean, by = "participant")
```

```
## Warning in left_join(., rt_clean, by = "participant"): Detected an unexpected many-to-
many relationship between `x` and `y`.
## i Row 1 of `x` matches multiple rows in `y`.
## i Row 1 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```r
#model prerequisites
hist(df_joined_m2$RT, breaks = 50) #it looks skewed: we must log-transform
```



```r
df_joined_m2 <- df_joined_m2 %>%
  dplyr::mutate(
    logRT = log(RT)
  )
hist(df_joined_m2$logRT, breaks = 50) #looks great we can plot
```



```
summary(df_joined_m2[, c("delay_z", "in_z", "hp_z")])
```

```
##       delay_z.V1           in_z.V1             hp_z.V1
##  Min.   :-1.3029125   Min.   :-1.5246130   Min.   :-1.8628343
##  1st Qu.:-0.8649587   1st Qu.:-1.0250489   1st Qu.:-0.8755321
##  Median :-0.1806559   Median :-0.2757028   Median : 0.1117701
##  Mean   : 0.0000000   Mean   :-0.1117505   Mean   :-0.0310864
##  3rd Qu.: 0.7773680   3rd Qu.: 0.7234253   3rd Qu.: 0.7699715
##  Max.   : 1.8722525   Max.   : 2.0972265   Max.   : 1.9218240
```

```
df_joined_m2 <- df_joined_m2 %>%
  dplyr::mutate(
    delay_z = as.numeric(delay_z),
    in_z    = as.numeric(in_z),
    hp_z    = as.numeric(hp_z)
  )

summary(df_joined_m2[, c("delay_z", "in_z", "hp_z")])
```

```
##     delay_z             in_z               hp_z
##  Min.   :-1.3029   Min.   :-1.5246   Min.   :-1.86283
##  1st Qu.:-0.8650   1st Qu.:-1.0250   1st Qu.:-0.87553
##  Median :-0.1807   Median :-0.2757   Median : 0.11177
##  Mean   : 0.0000   Mean   :-0.1118   Mean   :-0.03109
##  3rd Qu.: 0.7774   3rd Qu.: 0.7234   3rd Qu.: 0.76997
##  Max.   : 1.8723   Max.   : 2.0972   Max.   : 1.92182
```

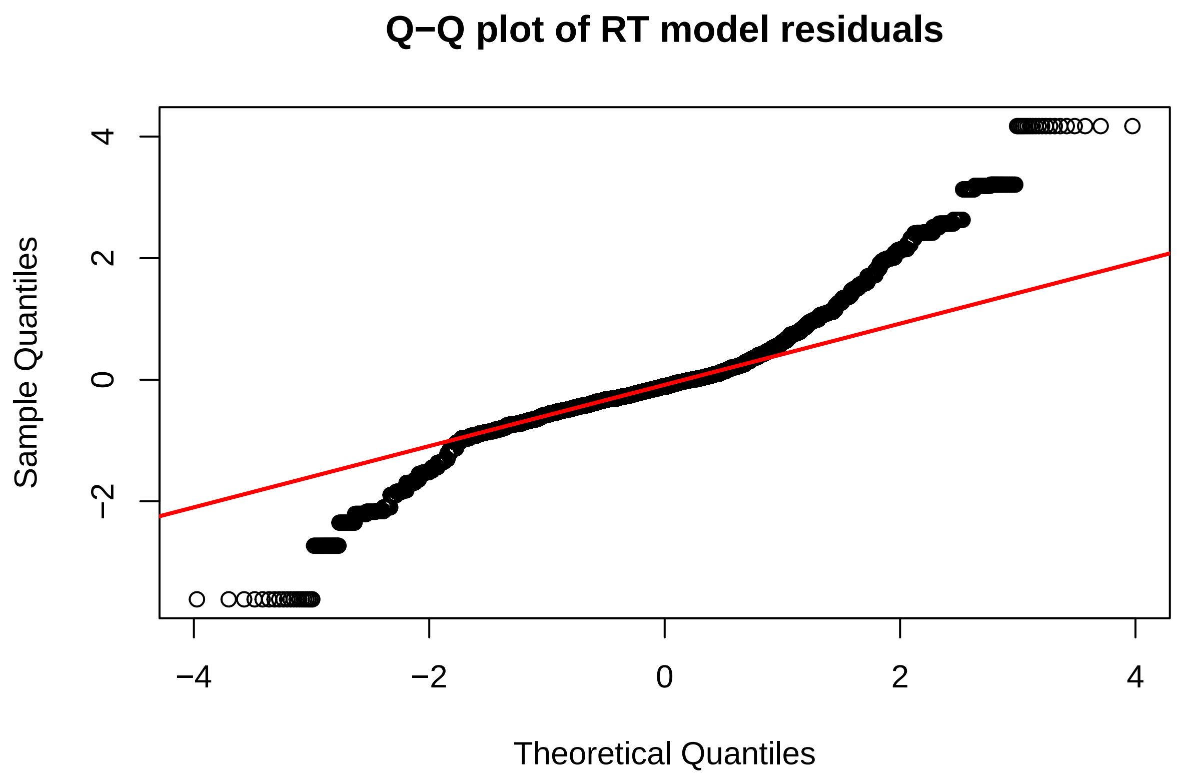
```
#check correlation between inattention & hyperactivity
cor(df_joined_m2$in_z, df_joined_m2$hp_z, use = "complete.obs")
```

```
## [1] 0.6626363
```

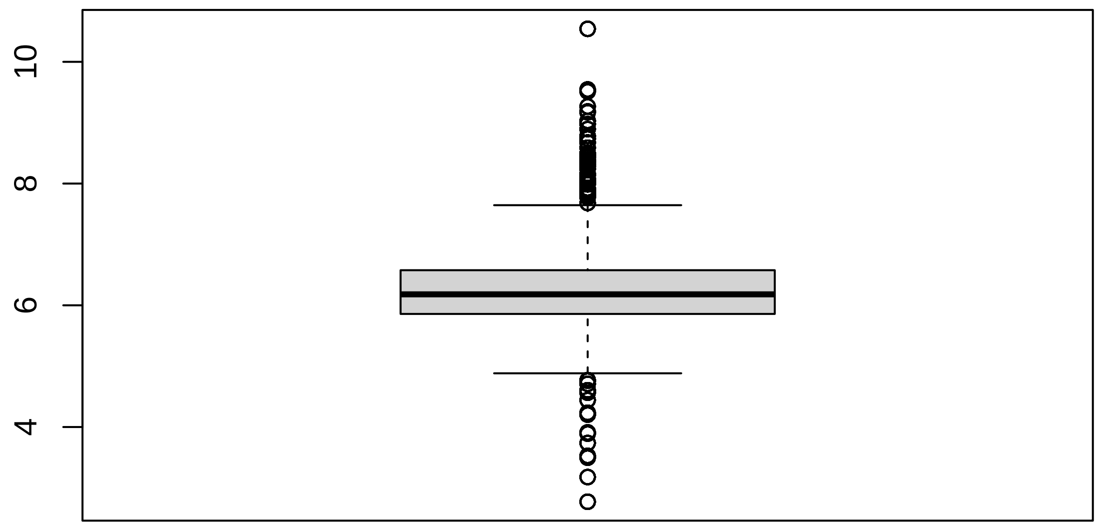
```
#model
library(lme4)

df_rt_model <- df_joined_m2 %>%
  dplyr::filter(RT > 0) %>%   # removes the 100 invalid trials
  dplyr::mutate(logRT = log(RT))

rt_model <- lmer(
  logRT ~ delay_z * in_z + delay_z * hp_z + (1 | participant),
  data = df_rt_model,
  REML = TRUE
)
summary(rt_model)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: logRT ~ delay_z * in_z + delay_z * hp_z + (1 | participant)
##    Data: df_rt_model
##
## REML criterion at convergence: 33799.6
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.5546 -0.5349 -0.1423  0.3217  5.2598
##
## Random effects:
##  Groups      Name        Variance Std.Dev.
##  participant (Intercept) 0.1007   0.3174
##  Residual                0.6290   0.7931
## Number of obs: 14160, groups:  participant, 36
##
## Fixed effects:
##                Estimate Std. Error t value
## (Intercept)   6.267e+00  5.384e-02 116.385
## delay_z      -9.891e-17  6.725e-03   0.000
## in_z         -9.367e-02  7.492e-02  -1.250
## hp_z          4.683e-02  6.835e-02   0.685
## delay_z:in_z -1.231e-15  9.374e-03   0.000
## delay_z:hp_z  1.033e-15  8.564e-03   0.000
##
## Correlation of Fixed Effects:
##             (Intr) dely_z in_z   hp_z   dly_z:n_
## delay_z      0.000
## in_z         0.136  0.000
## hp_z        -0.068  0.000 -0.660
## delay_z:n_z  0.000  0.126  0.000  0.000
## dely_z:hp_z  0.000 -0.062  0.000  0.000 -0.664
```

```r
#check for residual normality

qqnorm(residuals(rt_model), main = "Q-Q plot of RT model residuals")
qqline(residuals(rt_model), col = "red", lwd = 2)
```



```r
#central residuals are roughly normal, tails deviate a bit
boxplot(df_rt_model$logRT)
```



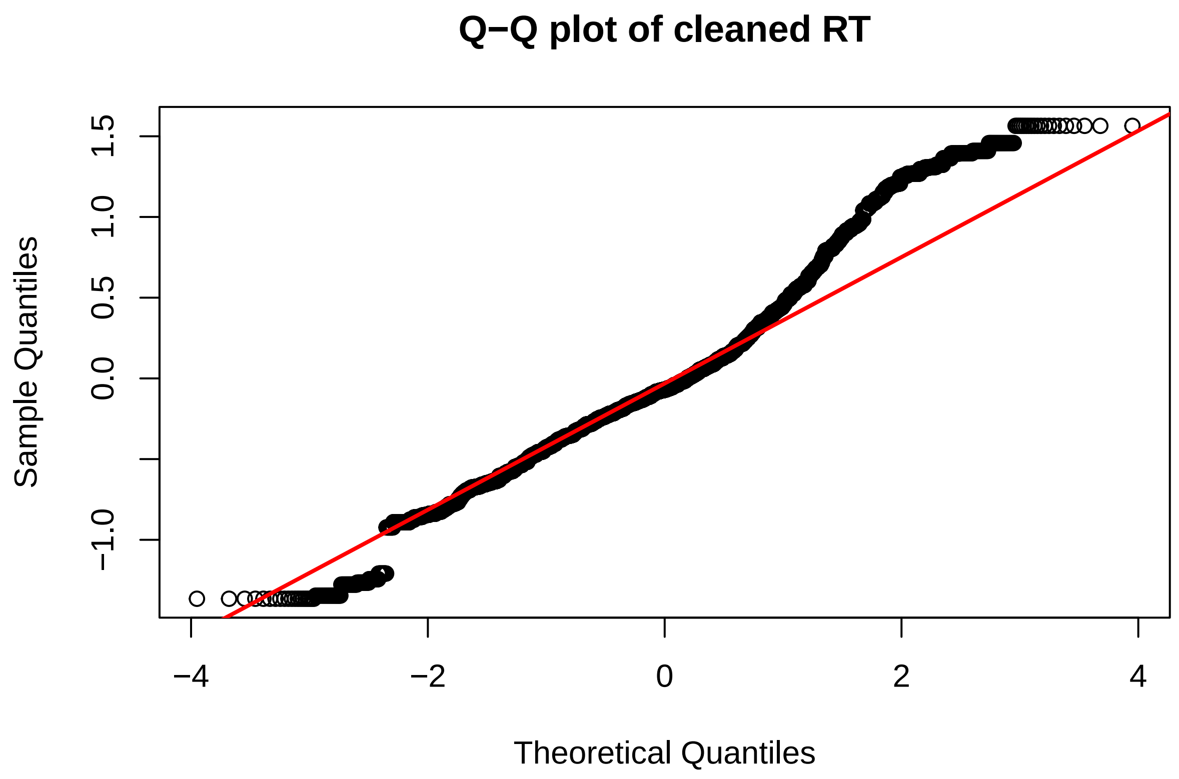
```
boxplot_stats <- boxplot.stats(df_rt_model$logRT)
extreme_RT <- df_joined_m2$logRT[df_joined_m2$logRT %in% boxplot_stats$out]
length(extreme_RT)  # see how many
```

```
## [1] 1340
```

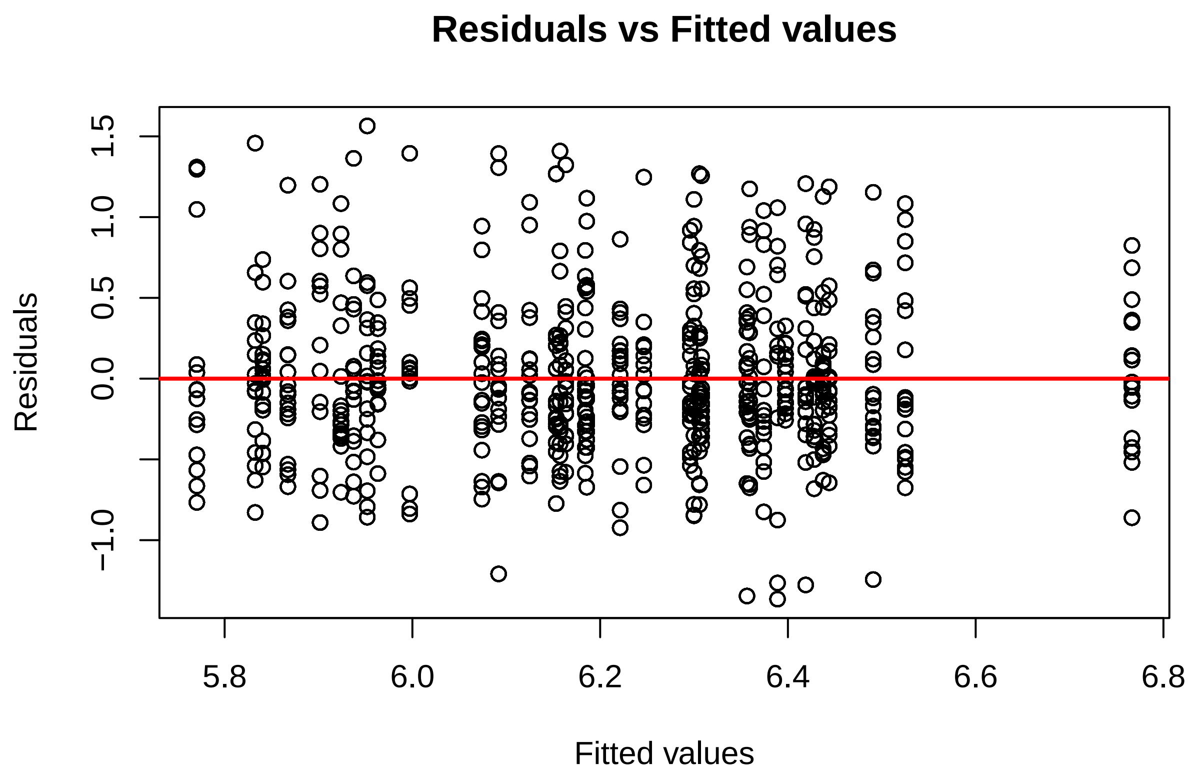
```
#we see heavy tailed residuals on the qqplot: we can try to remove extreme RTs
df_rt_model_clean <- df_rt_model %>%
  dplyr::filter(!logRT %in% extreme_RT)

rt_model2_clean <- lmer(
  logRT ~ delay_z * in_z + delay_z * hp_z + (1 | participant),
  data = df_rt_model_clean,
  REML = TRUE
)
summary(rt_model2_clean)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: logRT ~ delay_z * in_z + delay_z * hp_z + (1 | participant)
##    Data: df_rt_model_clean
##
## REML criterion at convergence: 18300.4
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.7842 -0.6021 -0.1416  0.4746  3.1910
##
## Random effects:
##  Groups      Name        Variance Std.Dev.
##  participant (Intercept) 0.05822  0.2413
##  Residual                0.24038  0.4903
## Number of obs: 12820, groups:  participant, 36
##
## Fixed effects:
##                 Estimate Std. Error t value
## (Intercept)    6.196e+00  4.085e-02 151.697
## delay_z       -2.785e-16  4.371e-03   0.000
## in_z          -3.213e-02  5.683e-02  -0.565
## hp_z           1.898e-02  5.185e-02   0.366
## delay_z:in_z  -8.007e-16  6.009e-03   0.000
## delay_z:hp_z   4.766e-16  5.508e-03   0.000
##
## Correlation of Fixed Effects:
##              (Intr) dely_z in_z   hp_z   dly_z:n_
## delay_z       0.000
## in_z          0.136  0.000
## hp_z         -0.069  0.000 -0.660
## delay_z:n_z   0.000  0.134  0.000  0.000
## dely_z:hp_z   0.000 -0.086  0.000  0.000 -0.662
```

```
#Chechking model assumptions again
qqnorm(residuals(rt_model2_clean), main = "Q-Q plot of cleaned RT")
qqline(residuals(rt_model2_clean), col = "red", lwd = 2)
```



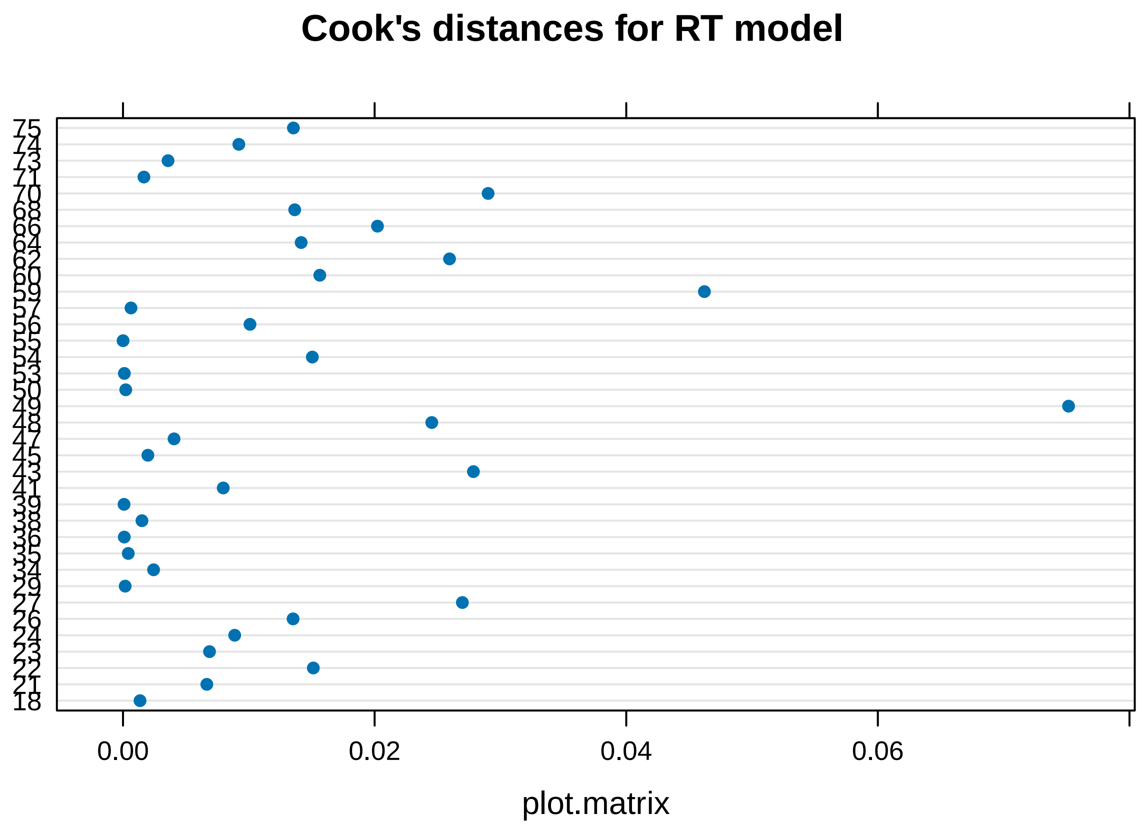
```r
#looks much better now

#Homoscedasticity
plot(fitted(rt_model2_clean), residuals(rt_model2_clean),
     xlab = "Fitted values",
     ylab = "Residuals",
     main = "Residuals vs Fitted values")
abline(h = 0, col = "red", lwd = 2)
```



```
#no clear pattern: homoscedasticity assumption met

#Influential participants
library(influence.ME)
```

```
## Warning: package 'influence.ME' was built under R version 4.4.3

##
## Attaching package: 'influence.ME'

## The following object is masked from 'package:stats':
##
##     influence
```

```
infl_rt <- influence(rt_model2_clean, group = "participant")
plot(infl_rt, which = "cook", main = "Cook's distances for RT model")
```



```
#One observation showed elevated Cook’s distance. Removing it did not materially change the results, so

#random effects variance (justification)
VarCorr(rt_model2_clean) #Participant-level variability in baseline reaction time justified the inclusi
```

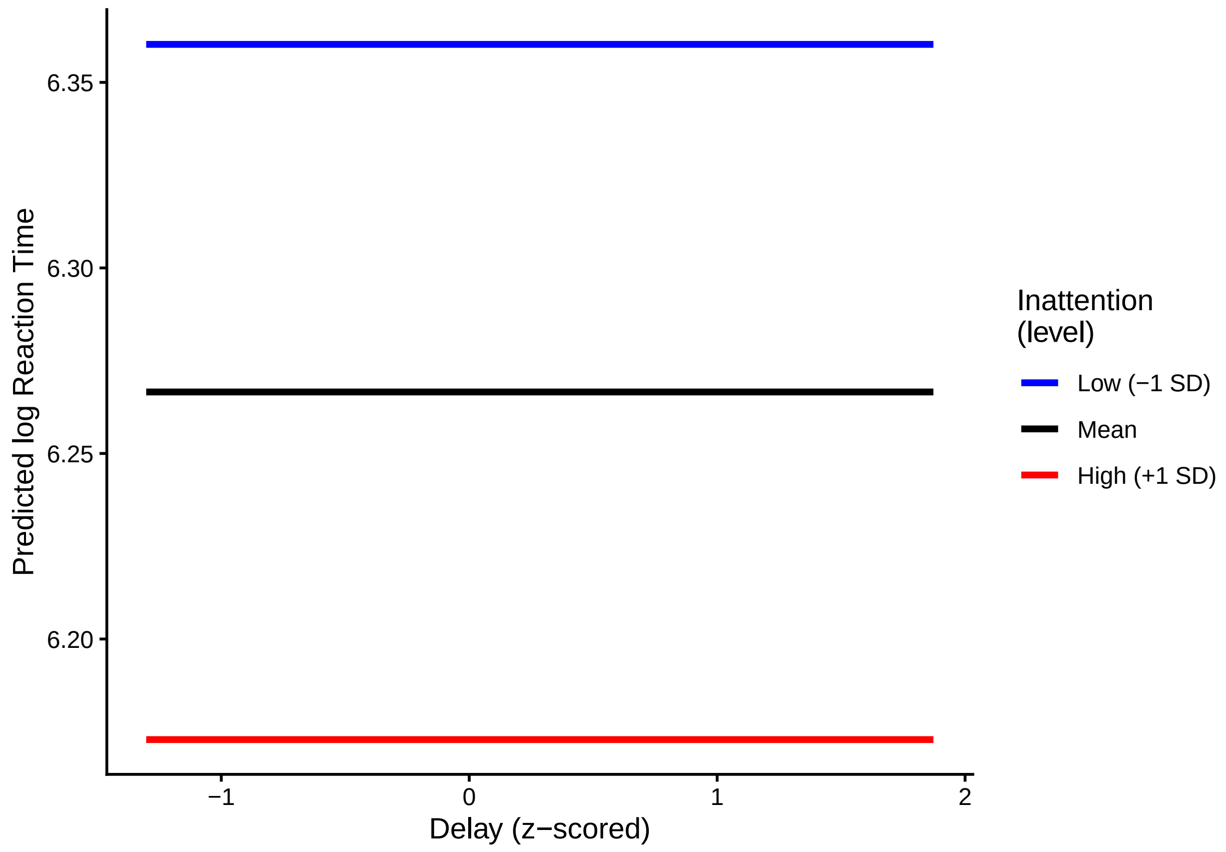
```
##  Groups      Name        Std.Dev.
##  participant (Intercept) 0.24128
##  Residual                0.49028
```

```
#multicollinearity
library(performance)

check_collinearity(rt_model2_clean) #Multicollinearity was assessed using variance inflation factors, a
```

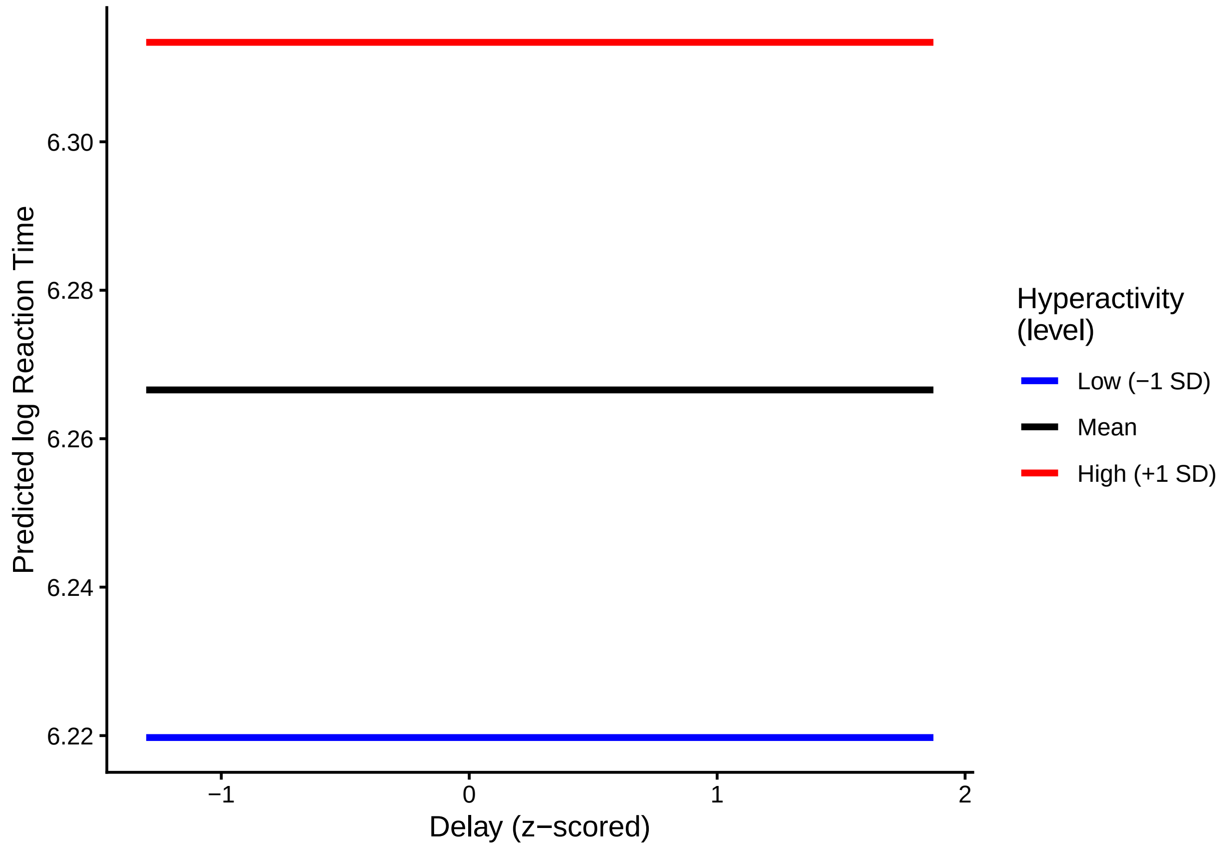
```
## # Check for Multicollinearity
##
## Low Correlation
##
##           Term  VIF   VIF 95% CI adj. VIF Tolerance Tolerance 95% CI
##        delay_z 1.02 [1.01, 1.05]     1.01      0.98     [0.95, 0.99]
##           in_z 1.77 [1.73, 1.82]     1.33      0.56     [0.55, 0.58]
##           hp_z 1.77 [1.73, 1.82]     1.33      0.56     [0.55, 0.58]
##  delay_z:in_z 1.80 [1.76, 1.85]     1.34      0.56     [0.54, 0.57]
##  delay_z:hp_z 1.78 [1.74, 1.83]     1.33      0.56     [0.55, 0.58]
```

```r
#plot
library(ggplot2)

# prediction grid
pred_in <- expand.grid(
  delay_z = seq(
    from = min(df_rt_model_clean$delay_z, na.rm = TRUE),
    to   = max(df_rt_model_clean$delay_z, na.rm = TRUE),
    length.out = 100
  ),
  in_z = c(-1, 0, 1),    # low, mean, high inattention
  hp_z = 0              # hold hyperactivity constant
)

# predicted values (fixed effects only)
pred_in$logRT_hat <- predict(
  rt_model,
  newdata = pred_in,
  re.form = NA
)

# plot
ggplot(pred_in, aes(x = delay_z, y = logRT_hat, color = factor(in_z))) +
  geom_line(linewidth = 1.2) +
  labs(
    x = "Delay (z-scored)",
    y = "Predicted log Reaction Time",
    color = "Inattention\n(level)",
    titel = "Predicted log RT by Delay and Inattention"
  ) +
  scale_color_manual(
    values = c("blue", "black", "red"),
    labels = c("Low (-1 SD)", "Mean", "High (+1 SD)")
  ) +
  theme_classic()
```

```
## Ignoring unknown labels:
## * titel : "Predicted log RT by Delay and Inattention"
```



```
pred_hp <- expand.grid(
  delay_z = seq(
    from = min(df_rt_model_clean$delay_z, na.rm = TRUE),
    to   = max(df_rt_model_clean$delay_z, na.rm = TRUE),
    length.out = 100
  ),
  hp_z = c(-1, 0, 1),    # low, mean, high hyperactivity
  in_z = 0              # hold inattention constant
)

pred_hp$logRT_hat <- predict(
  rt_model,
  newdata = pred_hp,
  re.form = NA
)

ggplot(pred_hp, aes(x = delay_z, y = logRT_hat, color = factor(hp_z))) +
  geom_line(linewidth = 1.2) +
  labs(
    x = "Delay (z-scored)",
    y = "Predicted log Reaction Time",
    color = "Hyperactivity\n(level)"
  ) +
  scale_color_manual(
    values = c("blue", "black", "red"),
    labels = c("Low (-1 SD)", "Mean", "High (+1 SD)")
  ) +
  theme_classic()
```



A linear mixed-effects model predicting log-transformed reaction times showed no main effect of delay, inattentive symptoms, or hyperactive symptoms. Additionally, no interactions between delay and ADHD symptom dimensions were observed. Participant-level random intercepts accounted for individual differences in baseline reaction time.

If you want to be extra rigorous:

Model diagnostics indicated no violations of model assumptions. One observation showed elevated influence, but sensitivity analyses indicated that retaining it did not alter the results. All continuous predictors were z-standardized prior to analysis. Model diagnostics indicated approximately normal residuals and no major violations of homoscedasticity.

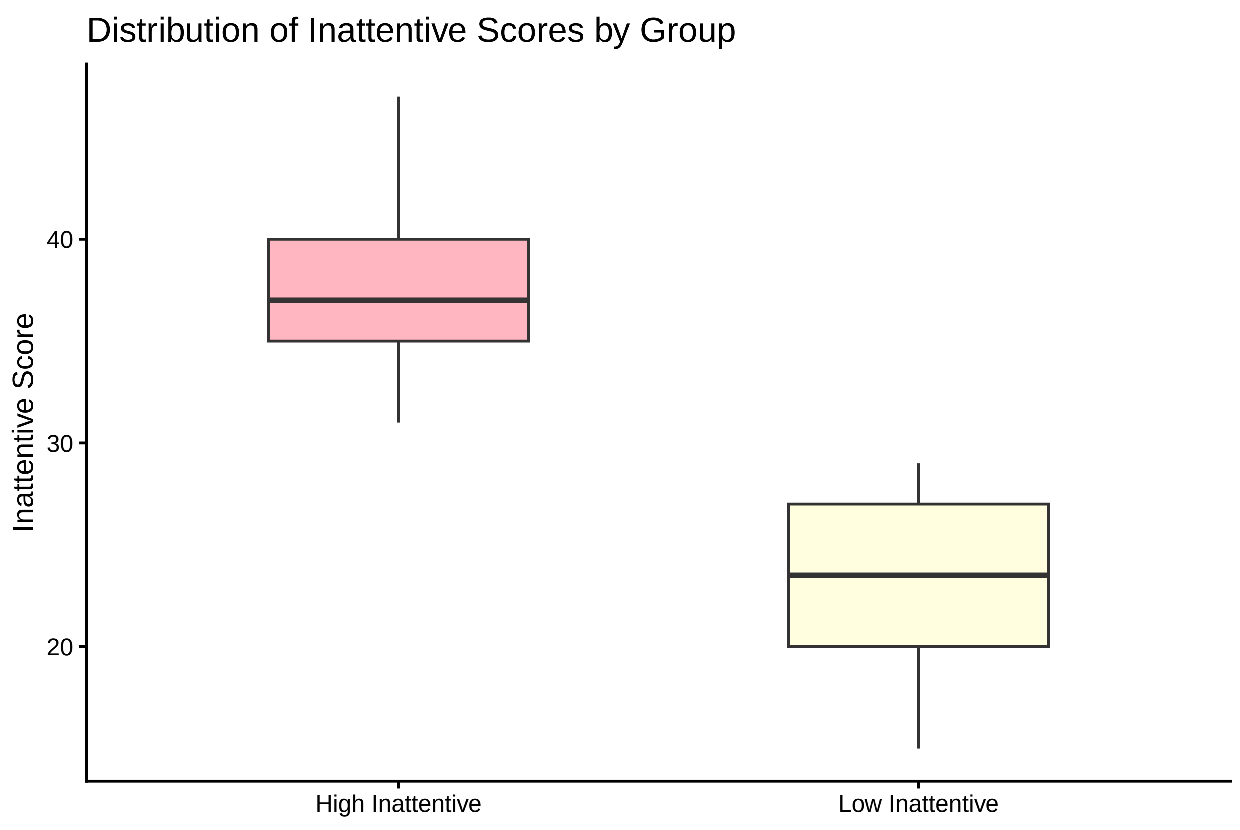
Predicted reaction times showed no effect of delay and no interaction with inattentive or hyperactive symptoms, as indicated by overlapping regression lines.Delay does not affect reaction time, and ADHD symptom levels do not change that relationship.

```
# Median split of inattentive scores
med_in <- median(questionnaire$in_score, na.rm = TRUE)

# Create ad hoc groups based on in_score
questionnaire$in_group <- ifelse(questionnaire$in_score <= med_in,
                                 "Low Inattentive",
                                 "High Inattentive")
```

```r
# Optional: check group distribution
table(questionnaire$in_group)
```

```
##
## High Inattentive  Low Inattentive
##               25               28
```

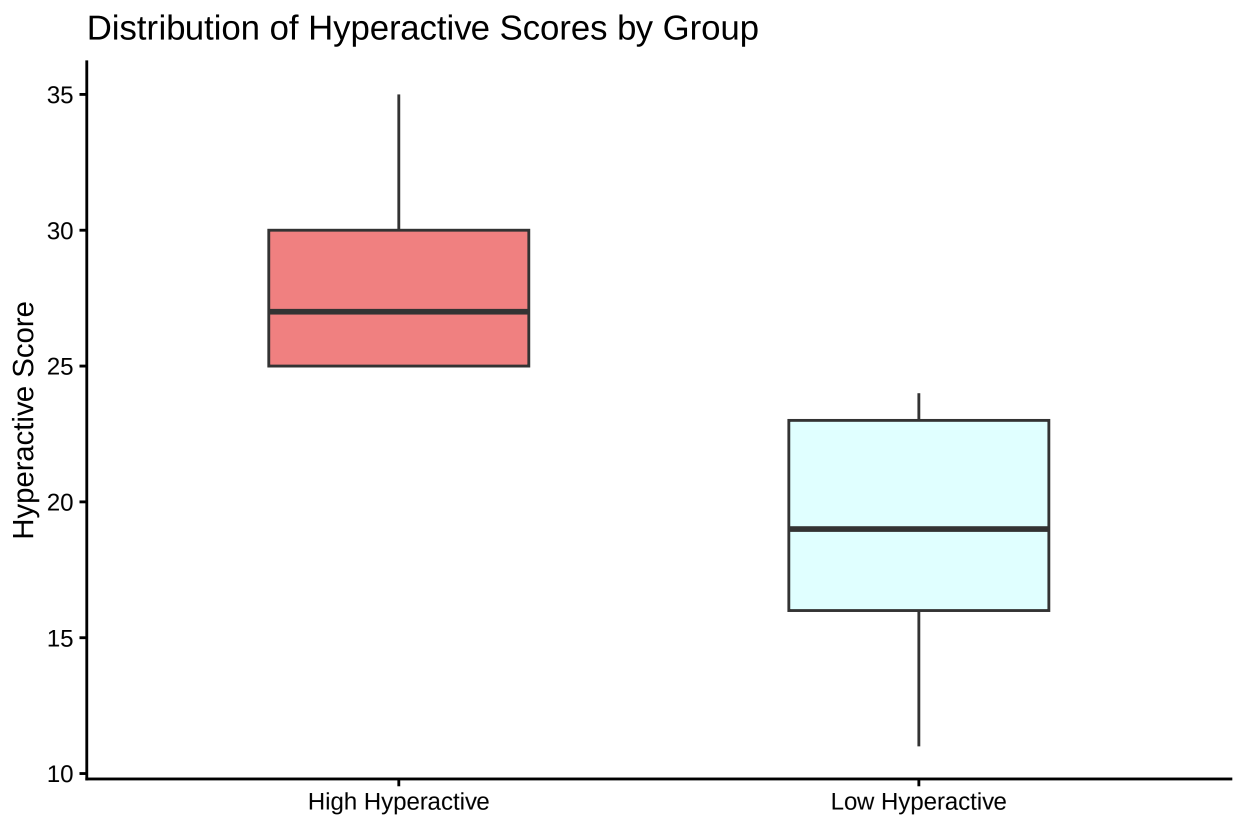
```r
# Compute summary stats per group
ranges_in <- tapply(questionnaire$in_score, questionnaire$in_group, summary)

# Create dataframe for plotting
df_summary_in <- data.frame(
  group = c("High Inattentive", "Low Inattentive"),
  ymin  = c(ranges_in[["High Inattentive"]]["Min."], ranges_in[["Low Inattentive"]]["Min."]),
  lower = c(ranges_in[["High Inattentive"]]["1st Qu."], ranges_in[["Low Inattentive"]]["1st Qu."]),
  middle= c(ranges_in[["High Inattentive"]]["Median"], ranges_in[["Low Inattentive"]]["Median"]),
  upper = c(ranges_in[["High Inattentive"]]["3rd Qu."], ranges_in[["Low Inattentive"]]["3rd Qu."]),
  ymax  = c(ranges_in[["High Inattentive"]]["Max."], ranges_in[["Low Inattentive"]]["Max."])
)

# Plot
ggplot(df_summary_in, aes(x = group)) +
  geom_boxplot(
    aes(
      ymin = ymin,
      lower = lower,
      middle = middle,
      upper = upper,
      ymax = ymax,
      fill = group
    ),
    stat = "identity",
    width = 0.5
  ) +
  labs(
    x = "",
    y = "Inattentive Score",
    title = "Distribution of Inattentive Scores by Group"
  ) +
  theme_classic() +
  scale_fill_manual(values = c("lightpink", "lightyellow")) +
  theme(legend.position = "none")
```



```r
# Median split of hyperactive scores
med_hp <- median(questionnaire$hp_score, na.rm = TRUE)

# Create ad hoc groups based on hp_score
questionnaire$hp_group <- ifelse(questionnaire$hp_score <= med_hp,
                                  "Low Hyperactive",
                                  "High Hyperactive")

# Optional: check group distribution
table(questionnaire$hp_group)
```

```
##
## High Hyperactive  Low Hyperactive
##               25               28
```

```r
# Compute summary stats per group
ranges_hp <- tapply(questionnaire$hp_score, questionnaire$hp_group, summary)

# Create dataframe for plotting
df_summary_hp <- data.frame(
  group = c("High Hyperactive", "Low Hyperactive"),
  ymin  = c(ranges_hp[["High Hyperactive"]]["Min."], ranges_hp[["Low Hyperactive"]]["Min."]),
  lower = c(ranges_hp[["High Hyperactive"]]["1st Qu."], ranges_hp[["Low Hyperactive"]]["1st Qu."]),
  middle= c(ranges_hp[["High Hyperactive"]]["Median"], ranges_hp[["Low Hyperactive"]]["Median"]),
  upper = c(ranges_hp[["High Hyperactive"]]["3rd Qu."], ranges_hp[["Low Hyperactive"]]["3rd Qu."]),
  ymax  = c(ranges_hp[["High Hyperactive"]]["Max."], ranges_hp[["Low Hyperactive"]]["Max."])
)

# Plot
ggplot(df_summary_hp, aes(x = group)) +
  geom_boxplot(
    aes(
      ymin = ymin,
      lower = lower,
      middle = middle,
      upper = upper,
      ymax = ymax,
      fill = group
    ),
    stat = "identity",
    width = 0.5
  ) +
  labs(
    x = "",
    y = "Hyperactive Score",
    title = "Distribution of Hyperactive Scores by Group"
  ) +
  theme_classic() +
  scale_fill_manual(values = c("lightcoral", "lightcyan")) +
  theme(legend.position = "none")
```



#Mouse tracking

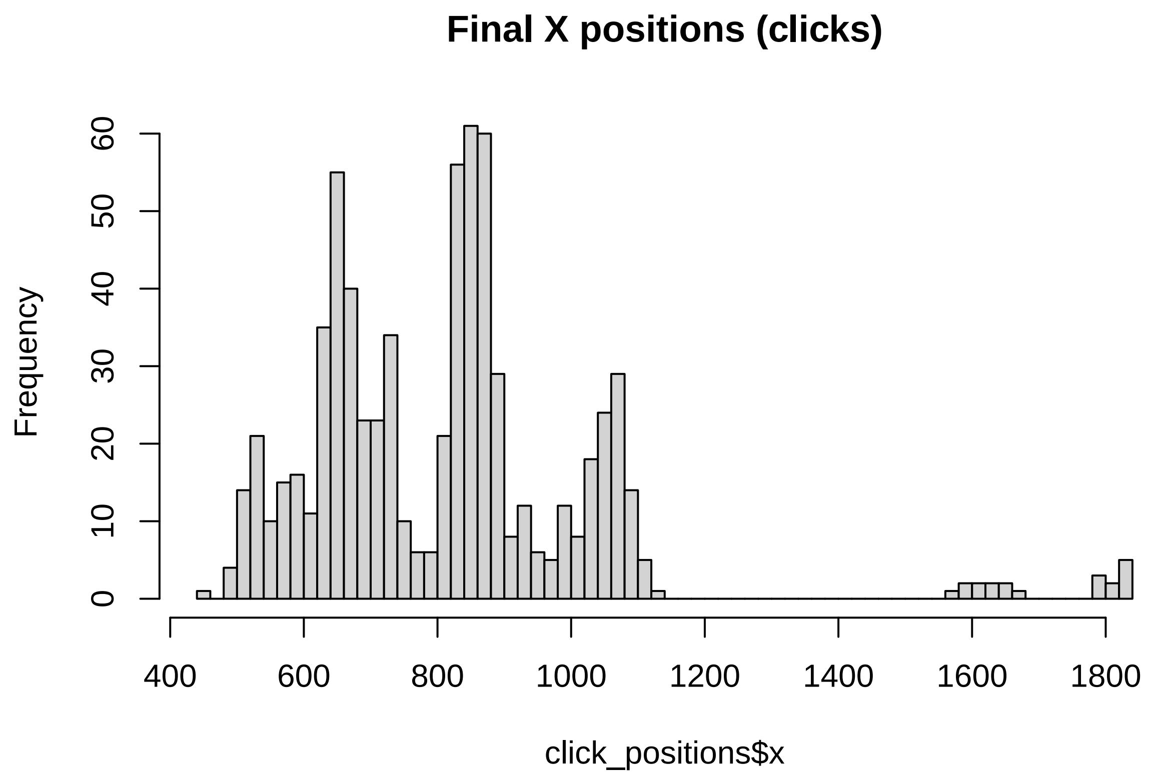
```r
#remove phone participants from mouse tracking data
mouse_tracking_clean <- mouse_tracking %>%
  filter(!is.na(x) & !is.na(y) & timestamp >= 0)

valid_participants <- unique(rt_clean$participant)

df_mouse <- mouse_tracking_clean %>%
  dplyr::filter(participant %in% valid_participants)

#get screen dimensions to normalize trajectories
summary(df_mouse$x)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   416.0   688.0   808.0   814.3   900.0  1840.0
```

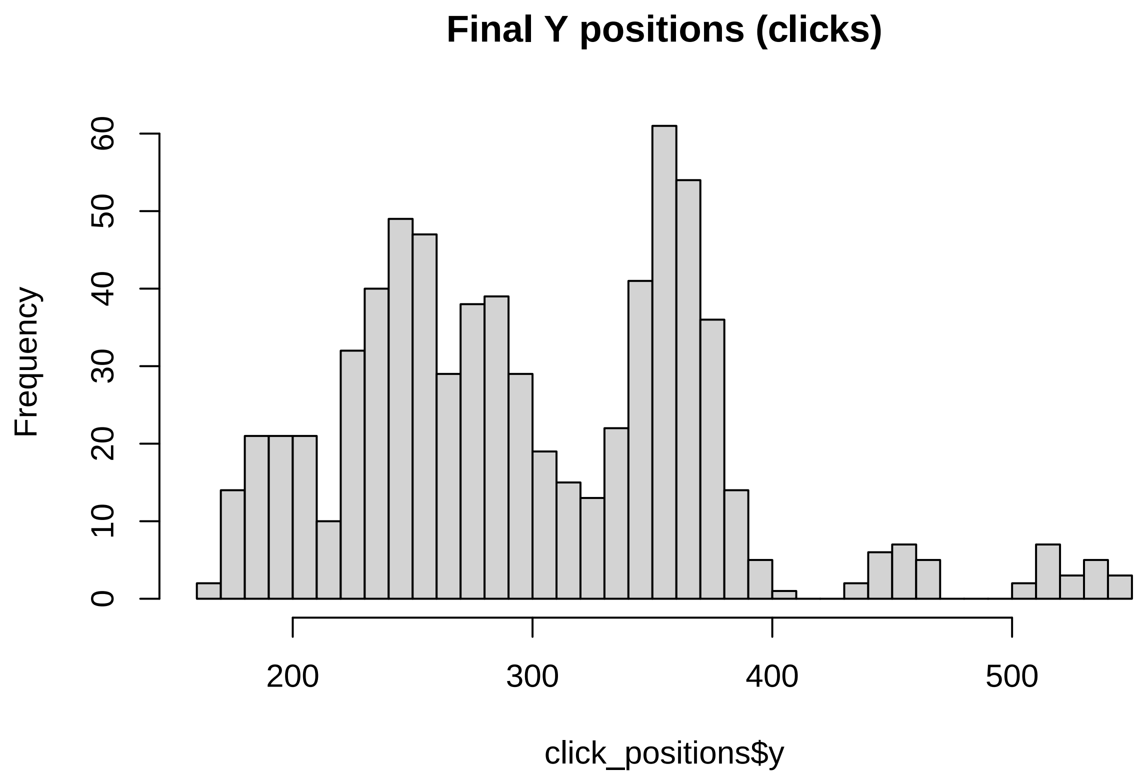
```r
summary(df_mouse$y)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   107.0   273.0   342.0   348.7   409.0   795.0
```

```r
click_positions <- df_mouse %>%
  group_by(participant, trial) %>%
  slice_tail(n = 1) %>%
  ungroup()

hist(click_positions$x, breaks = 50, main = "Final X positions (clicks)")
```



```r
hist(click_positions$y, breaks = 50, main = "Final Y positions (clicks)")
```



```r
#normalize by participant screen
df_mouse_norm <- df_mouse %>%
  group_by(participant) %>%
  mutate(
    screen_width = max(x),
    screen_height = max(y),
    x_norm = x / screen_width,
    y_norm = y / screen_height
  ) %>%
  ungroup()

#estimate button positions from clicks
estimate_buttons_from_clicks <- function(df_mouse_norm,
                                         button_width = 0.10,
                                         button_height = 0.06) {

  clicks <- df_mouse_norm %>%
    group_by(participant, trial) %>%
    arrange(timestamp) %>%
    slice_tail(n=1) %>%
    ungroup() %>%
    select(participant, trial, x_norm, y_norm)

  cat("Total clicks analyzed:", nrow(clicks), "\n")

  kmeans_result <- kmeans(clicks$x_norm, centers = 2)
  cluster_centers <- sort(kmeans_result$centers)
  left_button_x <- cluster_centers[1]
  right_button_x <- cluster_centers[2]

  clicks$cluster <- kmeans_result$cluster
  left_cluster_id <- which.min(kmeans_result$centers)
  right_cluster_id <- which.max(kmeans_result$centers)

  left_clicks <- clicks %>% filter(cluster == left_cluster_id)
  right_clicks <- clicks %>% filter(cluster == right_cluster_id)

  left_button_y <- median(left_clicks$y_norm)
  right_button_y <- median(right_clicks$y_norm)
  button_y <- mean(c(left_button_y, right_button_y))

  #print diagnostics
  cat("Left button center: (", round(left_button_x, 3), ",", round(button_y, 3), ")\n")
  cat("Right button center: (", round(right_button_x, 3), ",", round(button_y, 3), ")\n")
  cat("Left button clicks:", nrow(left_clicks), "\n")
  cat("Right button clicks:", nrow(right_clicks), "\n")

  #create button df with rectangles
  buttons <- data.frame(
    name = c("Together", "Delay"),
    x_center = c(left_button_x, right_button_x),
    y_center = c(button_y, button_y),
    x_left = c(left_button_x - button_width/2, right_button_x - button_width/2),
    x_right = c(left_button_x + button_width/2, right_button_x + button_width/2),
    y_top = c(button_y - button_height/2, button_y - button_height/2),
    y_bottom = c(button_y + button_height/2, button_y + button_height/2)
  )

  return(buttons)

}

buttons <- estimate_buttons_from_clicks(df_mouse_norm)
```

```
## Total clicks analyzed: 713
## Left button center: ( 0.719 , 0.542 )
## Right button center: ( 0.942 , 0.542 )
## Left button clicks: 392
## Right button clicks: 321
```

```r
#joining ADHD groups into mouse tracking data
df_mouse_grouped <- df_mouse_norm %>%
  left_join(
    questionnaire %>% select(participant, in_group, hp_group),
    by = "participant"
  )

interpolate_trajectory <- function(df, n_points = 101) {
  # Need at least 2 points to interpolate
  if (nrow(df) < 2) {
    return(data.frame(time_norm = numeric(0), x_norm = numeric(0), y_norm = numeric(0)))
  }

  # Create evenly spaced timepoints from 0 to 1
  standard_time <- seq(0, 1, length.out = n_points)

  # Normalize time within this trajectory
  df <- df %>%
    arrange(timestamp) %>%
    mutate(time_norm = (timestamp - min(timestamp)) / (max(timestamp) - min(timestamp)))

  # Check if all timestamps are the same (division by zero)
  if (all(is.na(df$time_norm)) || max(df$time_norm, na.rm = TRUE) == 0) {
    return(data.frame(time_norm = numeric(0), x_norm = numeric(0), y_norm = numeric(0)))
  }

  # Remove any duplicate time points
  df <- df %>%
    distinct(time_norm, .keep_all = TRUE)

  # Need at least 2 unique points
  if (nrow(df) < 2) {
    return(data.frame(time_norm = numeric(0), x_norm = numeric(0), y_norm = numeric(0)))
  }

  # Interpolate x and y to standard timepoints
  x_interp <- approx(df$time_norm, df$x_norm, xout = standard_time, rule = 2)$y
  y_interp <- approx(df$time_norm, df$y_norm, xout = standard_time, rule = 2)$y

  return(data.frame(
    time_norm = standard_time,
    x_norm = x_interp,
    y_norm = y_interp
  ))
}

# Interpolate all trajectories
df_mouse_interpolated <- df_mouse_grouped %>%
  group_by(participant, trial) %>%
  group_modify(~ interpolate_trajectory(.x, n_points = 101)) %>%
  ungroup() %>%
  filter(!is.na(x_norm) & !is.na(y_norm))

# Check diagnostics
cat("Original trajectories:",
    df_mouse_grouped %>% distinct(participant, trial) %>% nrow(), "\n")
```

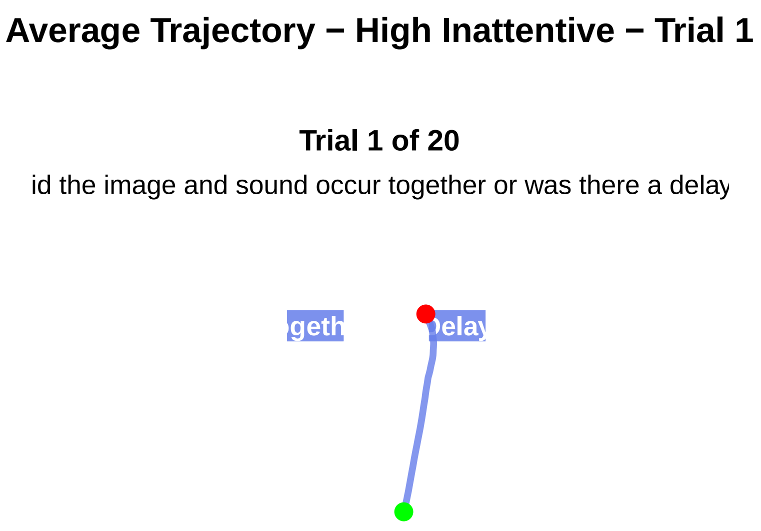
```
## Original trajectories: 713
```

```r
cat("Successfully interpolated:",
    df_mouse_interpolated %>% distinct(participant, trial) %>% nrow(), "\n")
```

```
## Successfully interpolated: 708
```

```r
# Average trajectories
avg_trajectories_in <- df_mouse_interpolated %>%
  left_join(
    questionnaire %>% select(participant, in_group),
    by = "participant"
  ) %>%
  group_by(in_group, trial, time_norm) %>%
  summarise(
    x_mean = mean(x_norm, na.rm = TRUE),
    y_mean = mean(y_norm, na.rm = TRUE),
    .groups = "drop"
  )

avg_trajectories_hp <- df_mouse_interpolated %>%
  left_join(
    questionnaire %>% select(participant, hp_group),
    by = "participant"
  ) %>%
  group_by(hp_group, trial, time_norm) %>%
  summarise(
    x_mean = mean(x_norm, na.rm = TRUE),
    y_mean = mean(y_norm, na.rm = TRUE),
    .groups = "drop"
  )


#function to plot one trial trajectory
plot_trial <- function(df, buttons, group_col, group_value, trial_n) {

  # Filter data for specific group and trial
  plot_data <- df %>%
    filter(!!sym(group_col) == group_value, trial == trial_n)

  # FLIP Y coordinates to match screen layout (1-y_norm)
  plot_data <- plot_data %>%
    mutate(
      x_mean = x_mean - 0.32,
      y_mean = 1 - y_mean + 0.20
      )

  # FLIP button Y coordinates too
  buttons_flipped <- buttons %>%
    mutate(
      x_center = x_center - 0.32,
      x_left = x_left - 0.32,
      x_right = x_right - 0.32,
      y_center = 1 - y_center + 0.20,
      y_top_new = 1 - y_bottom + 0.20,
      y_bottom_new = 1 - y_top + 0.20,
      y_top = y_top_new,
      y_bottom = y_bottom_new
    ) %>%
    select(-y_top_new, -y_bottom_new)

  p <- ggplot() +
    # Add button rectangles
    geom_rect(data = buttons_flipped,
              aes(xmin = x_left, xmax = x_right,
                  ymin = y_top, ymax = y_bottom),
              fill = "#667eea", alpha = 0.85, color = "white", linewidth = 1.2) +
    # Add button labels (SMALLER TEXT)
    geom_text(data = buttons_flipped,
              aes(x = x_center, y = y_center, label = name),
              color = "white", size = 3.5, fontface = "bold") +
    # Add average trajectory
    geom_path(data = plot_data,
              aes(x = x_mean, y = y_mean),
              color = "#667eea", linewidth = 1.2, alpha = 0.8) +
    # Add start and end points
    geom_point(data = plot_data %>% slice(1),
               aes(x = x_mean, y = y_mean),
               color = "green", size = 3, shape = 16) +
    geom_point(data = plot_data %>% slice(n()),
               aes(x = x_mean, y = y_mean),
               color = "red", size = 3, shape = 16) +
    # Add question text AT THE TOP (now at high Y values after flip)
    annotate("text", x = 0.5, y = 0.88,
             label = "Did the image and sound occur together or was there a delay?",
             size = 3.5, hjust = 0.5) +
    annotate("text", x = 0.5, y = 0.95,
             label = paste("Trial", trial_n, "of 20"),
             size = 4, fontface = "bold", hjust = 0.5) +
    # Styling
    coord_fixed(ratio = 1) +
    xlim(0, 1) +
    ylim(0, 1) +
    theme_minimal() +
    theme(
      panel.grid = element_blank(),
      axis.title = element_blank(),
      axis.text = element_blank(),
      axis.ticks = element_blank(),
      panel.background = element_rect(fill = "white", color = NA),
      plot.background = element_rect(fill = "white", color = NA),
      plot.title = element_text(hjust = 0.5, size = 13, face = "bold",
                                margin = margin(b = 10)),
      plot.margin = margin(t = 20, r = 20, b = 20, l = 20)  # Center the plot
    ) +
    labs(title = paste("Average Trajectory -", group_value, "- Trial", trial_n))

  return(p)
}

# Recreate all plots with corrected function
plots_in_group <- list()
```

```r
for (group_val in unique(avg_trajectories_in$in_group)) {
  for (trial_n in 1:20) {
    plot_name <- paste0("in_group_", group_val, "_trial_", trial_n)
    plots_in_group[[plot_name]] <- plot_trial(
      avg_trajectories_in,
      buttons,
      "in_group",
      group_val,
      trial_n
    )
  }
}

plots_hp_group <- list()
for (group_val in unique(avg_trajectories_hp$hp_group)) {
  for (trial_n in 1:20) {
    plot_name <- paste0("hp_group_", group_val, "_trial_", trial_n)
    plots_hp_group[[plot_name]] <- plot_trial(
      avg_trajectories_hp,
      buttons,
      "hp_group",
      group_val,
      trial_n
    )
  }
}

# View the corrected plot
plots_in_group[[1]]
```



```r
# Save all plots to files (optional)
library(ggplot2)

# Create directories
dir.create("plots/in_group", recursive = TRUE, showWarnings = FALSE)
dir.create("plots/hp_group", recursive = TRUE, showWarnings = FALSE)

# Save in_group plots
for (group_val in unique(avg_trajectories_in$in_group)) {
  for (trial_n in 1:20) {
    plot_name <- paste0("in_group_", group_val, "_trial_", trial_n)
    filename <- paste0("plots/in_group/", plot_name, ".png")
    ggsave(filename,
           plots_in_group[[plot_name]],
           width = 8,
           height = 8,
           dpi = 300)
  }
}

# Save hp_group plots
for (group_val in unique(avg_trajectories_hp$hp_group)) {
  for (trial_n in 1:20) {
    plot_name <- paste0("hp_group_", group_val, "_trial_", trial_n)
    filename <- paste0("plots/hp_group/", plot_name, ".png")
    ggsave(filename,
            plots_hp_group[[plot_name]],
            width = 8,
            height = 8,
            dpi = 300)
  }
}

cat("All plots created and saved!\n")
```

```
## All plots created and saved!
```

```r
cat("Total plots created:", length(plots_in_group) + length(plots_hp_group), "\n")
```

```
## Total plots created: 80
```

#Calculating the just noticable difference for each group (JND)

```r
#prepping data frame
correctness <- na.omit(correctness)
correctness <- correctness %>%
  left_join(
    questionnaire %>%
      select(participant, (ncol(questionnaire)-1):ncol(questionnaire)),
    by = "participant"
  )

#coding responses as 0 and 1 (delay 1, together 0)
correctness <- correctness %>%
  mutate(responses.user_response = if_else(responses.user_response == "delay", 1L, 0L))

#calculations
calculate_threshold <- function(df) {
  #fit logistic regression: detect delay as function of absolute SOA
  model <- glm(responses.user_response ~ responses.delay_ms,
               data = df,
               family = binomial(link = "logit"))

  #calculate thresholds at different detection levels
  #75% threshold: SOA where participant detects delay 75% of the time
  threshold_75 <- (log(3) - coef(model)[1] / coef(model)[2])

  #50% threshold: SOA where participant detects delay 50% of the time
  threshold_50 <- -coef(model)[1] / coef(model)[2]

  #slope: steeper = better discrimination
  slope <- coef(model)[2]

  #JND estimate (difference between 75% and 25% thresholds)
  jnd <- log(3) / abs(coef(model)[2])

  return(data.frame(
    threshold_50 = threshold_50,
    threshold_75 = threshold_75,
    jnd = jnd,
    slope = slope
  ))
}

#calculate thresholds for hyperactivity groups
cat("=== CALCULATING TEMPORAL SENSITIVITY BY GROUP ===\n\n")
```

```
## === CALCULATING TEMPORAL SENSITIVITY BY GROUP ===
```

```
thresholds_hp <- correctness %>%
  group_by(hp_group) %>%
  do(calculate_threshold(.)) %>%
  arrange(threshold_50)

cat("Hyperactivity Groups:\n")
```

```
## Hyperactivity Groups:
```

```
print(thresholds_hp)
```

```
## # A tibble: 2 x 5
## # Groups:   hp_group [2]
##   hp_group         threshold_50 threshold_75   jnd   slope
##   <chr>                   <dbl>        <dbl> <dbl>   <dbl>
## 1 High Hyperactive         121.         122. 149.  0.00736
## 2 Low Hyperactive          151.         152.  81.1 0.0135
```

```
#calculate thresholds for inattentive groups
thresholds_in <- correctness %>%
  group_by(in_group) %>%
  do(calculate_threshold(.)) %>%
  arrange(threshold_50)

cat("\nInattention Groups:\n")
```

```
##
## Inattention Groups:
```

```
print(thresholds_in)
```

```
## # A tibble: 2 x 5
## # Groups:   in_group [2]
##   in_group         threshold_50 threshold_75   jnd   slope
##   <chr>                   <dbl>        <dbl> <dbl>   <dbl>
## 1 High Inattentive         136.         137.  114. 0.00967
## 2 Low Inattentive          144.         145.  101. 0.0109
```

```r
# Interpretation
cat("\n=== INTERPRETATION ===\n")
```

```
##
## === INTERPRETATION ===
```

```r
cat("Threshold_50: SOA (ms) where participants detect delay 50% of the time\n")
```

```
## Threshold_50: SOA (ms) where participants detect delay 50% of the time
```

```r
cat("  - HIGHER threshold = WORSE temporal discrimination (more difficulty)\n")
```

```
##   - HIGHER threshold = WORSE temporal discrimination (more difficulty)
```

```r
cat("  - LOWER threshold = BETTER temporal discrimination (less difficulty)\n\n")
```

```
##   - LOWER threshold = BETTER temporal discrimination (less difficulty)
```

```r
cat("JND (Just Noticeable Difference): Range of uncertainty\n")
```

```
## JND (Just Noticeable Difference): Range of uncertainty
```

```r
cat("  - LARGER JND = MORE difficulty discriminating temporal order\n")
```

```
##   - LARGER JND = MORE difficulty discriminating temporal order
```

```r
cat("  - SMALLER JND = LESS difficulty discriminating temporal order\n\n")
```

```
##   - SMALLER JND = LESS difficulty discriminating temporal order
```

```r
cat("Slope: Steepness of psychometric curve\n")
```

```
## Slope: Steepness of psychometric curve
```

```r
cat("  - HIGHER slope = SHARPER discrimination (better performance)\n")
```

```
##   - HIGHER slope = SHARPER discrimination (better performance)
```

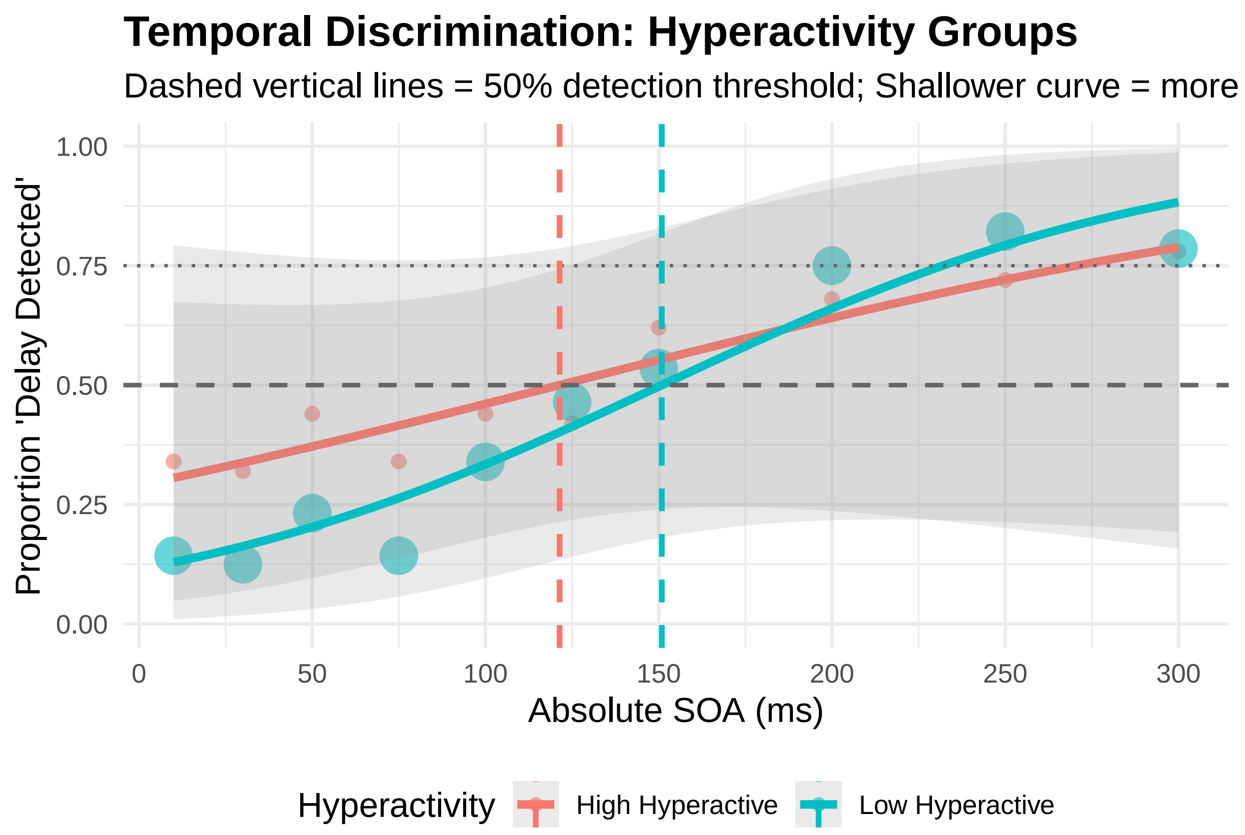
```r
cat("  - LOWER slope = SHALLOWER discrimination (worse performance)\n\n")
```

```
##   - LOWER slope = SHALLOWER discrimination (worse performance)
```

```r
# Summary statistics by group and SOA
summary_hp <- correctness %>%
  group_by(responses.delay_ms, hp_group) %>%
  summarise(
    prop_delay = mean(responses.user_response),
    se = sqrt(prop_delay * (1 - prop_delay) / n()),
    n = n(),
    .groups = "drop"
  )

summary_in <- correctness %>%
  group_by(responses.delay_ms, in_group) %>%
  summarise(
    prop_delay = mean(responses.user_response),
    se = sqrt(prop_delay * (1 - prop_delay) / n()),
    n = n(),
    .groups = "drop"
  )

# Visualization 1: Psychometric curves - Hyperactivity
p1 <- ggplot(summary_hp, aes(x = responses.delay_ms, y = prop_delay, color = hp_group)) +
  geom_point(aes(size = n), alpha = 0.6) +
  geom_smooth(method = "glm",
              method.args = list(family = "binomial"),
              se = TRUE, alpha = 0.2, linewidth = 1.5) +
  geom_hline(yintercept = 0.5, linetype = "dashed", color = "gray40", linewidth = 0.8) +
  geom_hline(yintercept = 0.75, linetype = "dotted", color = "gray40", linewidth = 0.6) +
  geom_vline(data = thresholds_hp,
             aes(xintercept = threshold_50, color = hp_group),
             linetype = "dashed", linewidth = 1) +
  scale_size_continuous(range = c(2, 6), guide = "none") +
  labs(
    title = "Temporal Discrimination: Hyperactivity Groups",
    subtitle = "Dashed vertical lines = 50% detection threshold; Shallower curve = more difficulty",
    x = "Absolute SOA (ms)",
    y = "Proportion 'Delay Detected'",
    color = "Hyperactivity"
  ) +
  scale_x_continuous(breaks = seq(0, 300, 50)) +
  scale_y_continuous(breaks = seq(0, 1, 0.25), limits = c(0, 1)) +
  theme_minimal(base_size = 13) +
  theme(legend.position = "bottom",
        plot.title = element_text(face = "bold"))

print(p1)
```

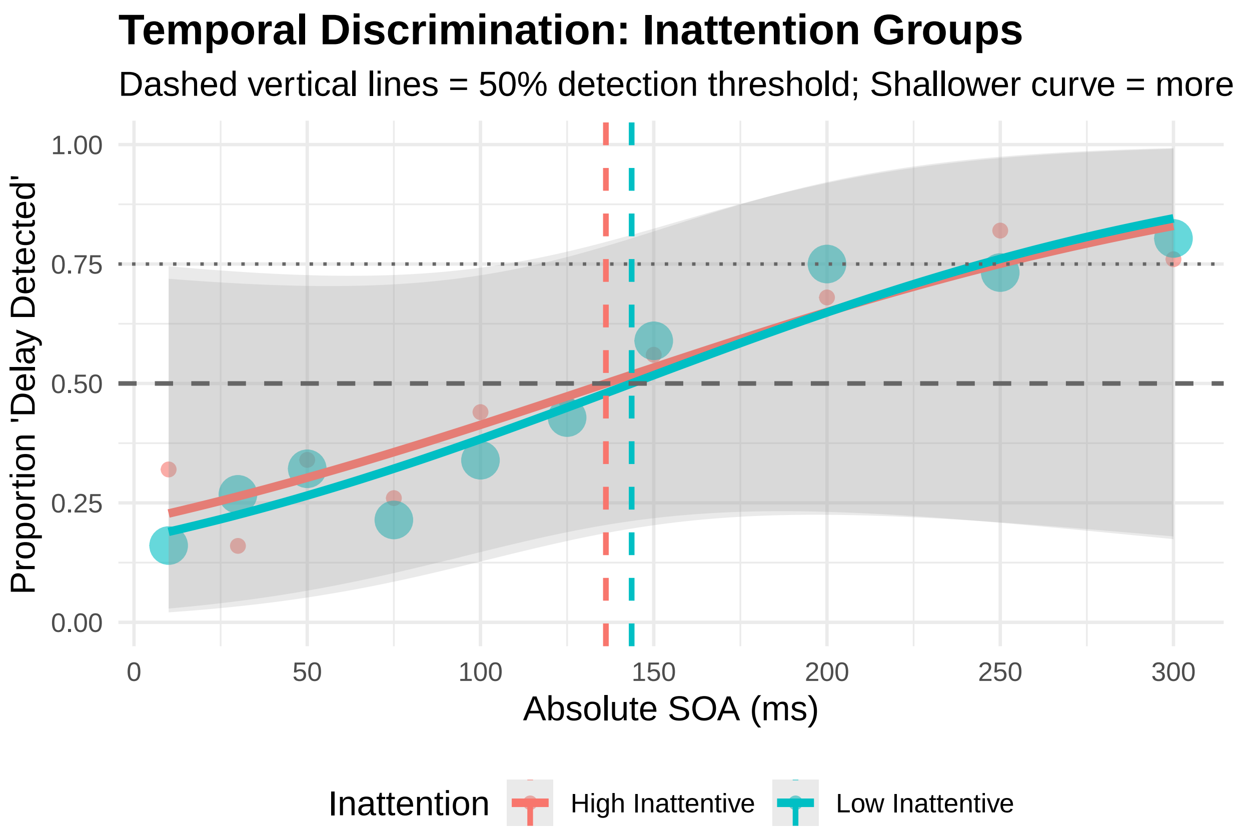
```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

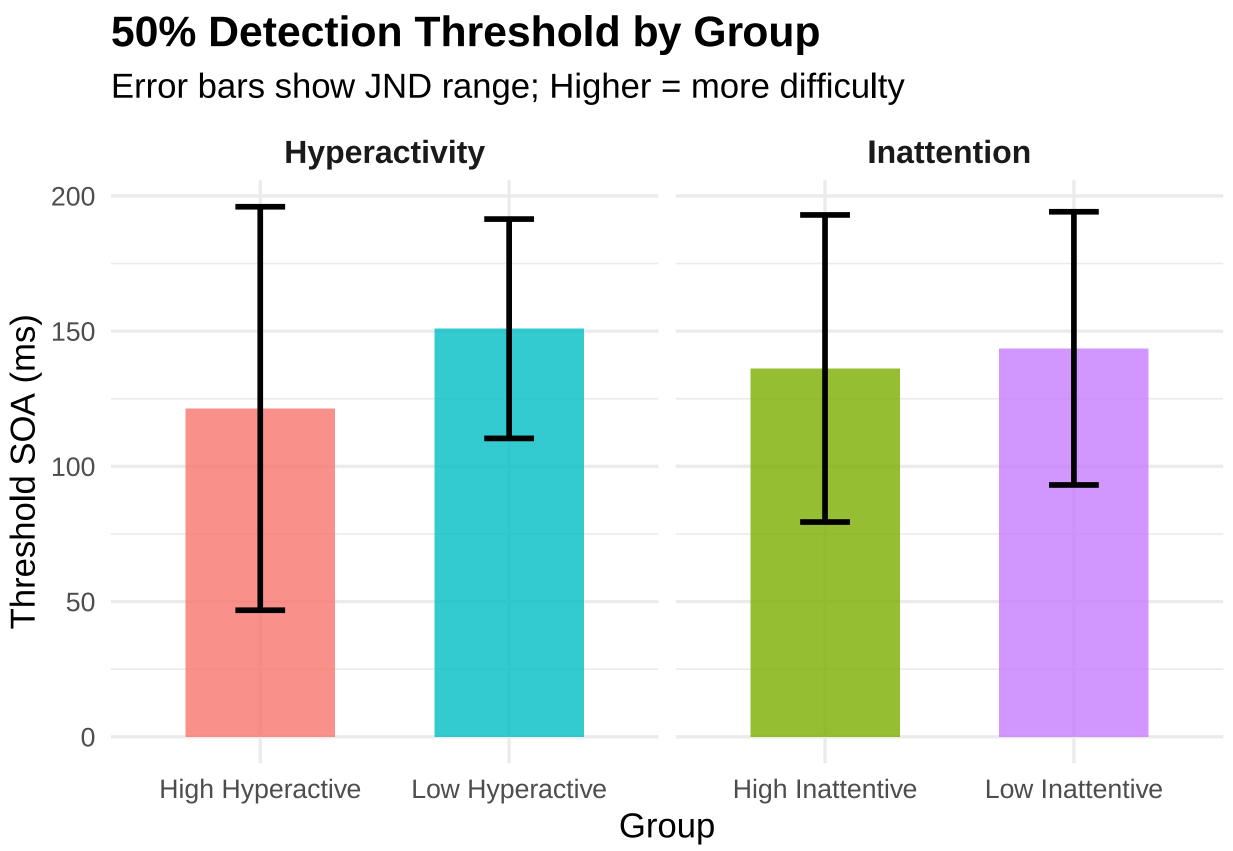


```r
# Visualization 2: Inattention groups
p2 <- ggplot(summary_in, aes(x = responses.delay_ms, y = prop_delay, color = in_group)) +
  geom_point(aes(size = n), alpha = 0.6) +
  geom_smooth(method = "glm",
              method.args = list(family = "binomial"),
              se = TRUE, alpha = 0.2, linewidth = 1.5) +
  geom_hline(yintercept = 0.5, linetype = "dashed", color = "gray40", linewidth = 0.8) +
  geom_hline(yintercept = 0.75, linetype = "dotted", color = "gray40", linewidth = 0.6) +
  geom_vline(data = thresholds_in,
             aes(xintercept = threshold_50, color = in_group),
             linetype = "dashed", linewidth = 1) +
  scale_size_continuous(range = c(2, 6), guide = "none") +
  labs(
    title = "Temporal Discrimination: Inattention Groups",
    subtitle = "Dashed vertical lines = 50% detection threshold; Shallower curve = more difficulty",
    x = "Absolute SOA (ms)",
    y = "Proportion 'Delay Detected'",
    color = "Inattention"
  ) +
  scale_x_continuous(breaks = seq(0, 300, 50)) +
  scale_y_continuous(breaks = seq(0, 1, 0.25), limits = c(0, 1)) +
  theme_minimal(base_size = 13) +
  theme(legend.position = "bottom",
        plot.title = element_text(face = "bold"))

print(p2)
```

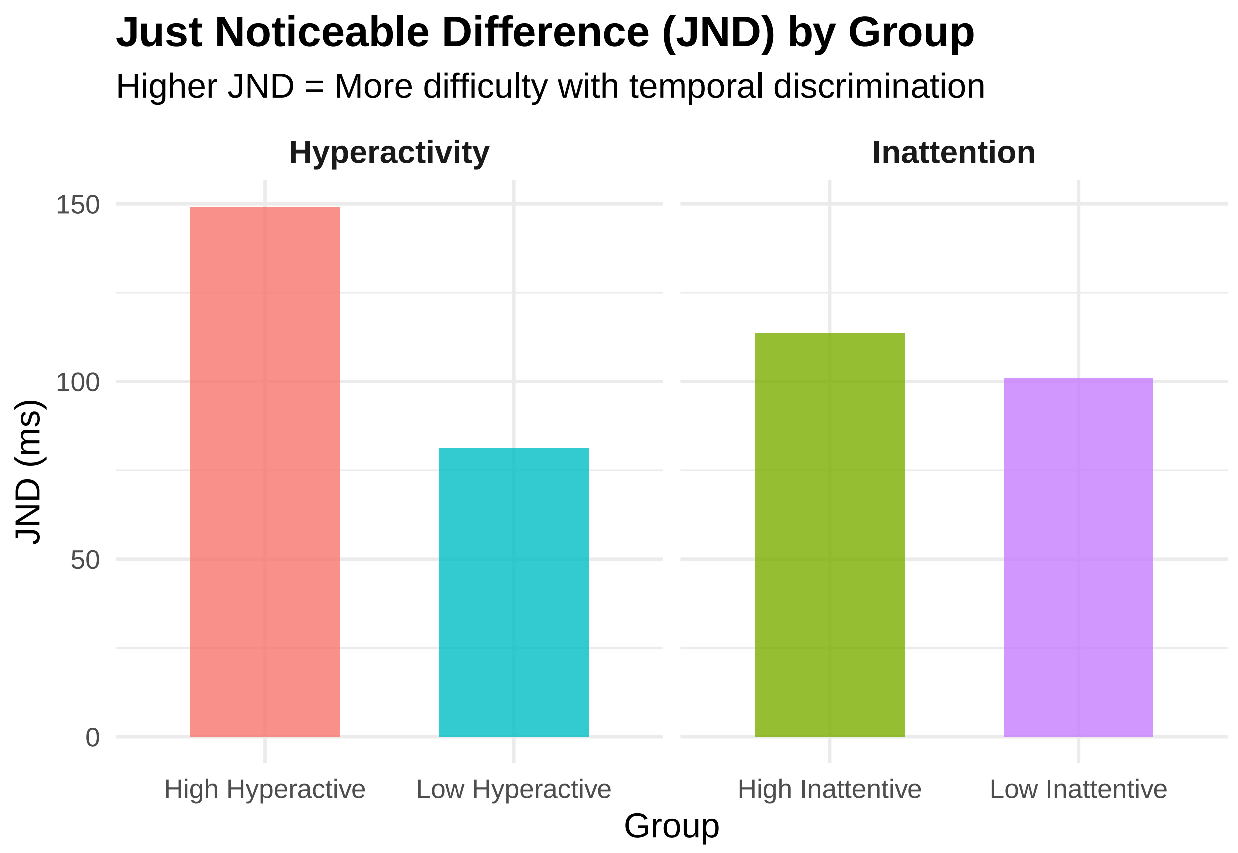
```
## `geom_smooth()` using formula = 'y ~ x'

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```



```r
# Visualization 3: Threshold comparison
thresholds_combined <- bind_rows(
  thresholds_hp %>% mutate(dimension = "Hyperactivity") %>% rename(group = hp_group),
  thresholds_in %>% mutate(dimension = "Inattention") %>% rename(group = in_group)
)

p3 <- ggplot(thresholds_combined, aes(x = group, y = threshold_50, fill = group)) +
  geom_col(alpha = 0.8, width = 0.6) +
  geom_errorbar(aes(ymin = threshold_50 - jnd/2, ymax = threshold_50 + jnd/2),
              width = 0.2, linewidth = 1) +
  facet_wrap(~dimension, scales = "free_x") +
  labs(
    title = "50% Detection Threshold by Group",
    subtitle = "Error bars show JND range; Higher = more difficulty",
    x = "Group",
    y = "Threshold SOA (ms)"
  ) +
  theme_minimal(base_size = 13) +
  theme(legend.position = "none",
        plot.title = element_text(face = "bold"),
        strip.text = element_text(face = "bold", size = 12))

print(p3)
```



```
# Visualization 4: JND comparison
p4 <- ggplot(thresholds_combined, aes(x = group, y = jnd, fill = group)) +
  geom_col(alpha = 0.8, width = 0.6) +
  facet_wrap(~dimension, scales = "free_x") +
  labs(
    title = "Just Noticeable Difference (JND) by Group",
    subtitle = "Higher JND = More difficulty with temporal discrimination",
    x = "Group",
    y = "JND (ms)"
  ) +
  theme_minimal(base_size = 13) +
  theme(legend.position = "none",
        plot.title = element_text(face = "bold"),
        strip.text = element_text(face = "bold", size = 12))

print(p4)
```



#p1 and p2:

Lines:

Solid colored lines (red/blue) = Fitted psychometric curves showing P(detect delay) as function of SOA Dashed horizontal line at 0.5 = 50% detection threshold (where participants are most uncertain) Dotted horizontal line at 0.75 = 75% detection threshold Dashed vertical lines (red/blue) = 50% detection threshold for each group (where their curve crosses 0.5)

Points:

Colored dots = Observed proportion of "delay detected" responses at each SOA Dot size = Number of trials/observations at that SOA for that group

Interpretation of your plot:

Low Hyperactive (blue): Steeper curve, crosses 50% around 150ms High Hyperactive (red): Shallower curve (more difficulty!), crosses 50% around 130ms The shallower red curve means High Hyperactive participants have worse temporal discrimination

#Statistical analysis jnd

```r
# Statistical testing at participant level
cat("\n=== STATISTICAL ANALYSIS ===\n\n")
```

```
##
## === STATISTICAL ANALYSIS ===
```

```r
# Calculate thresholds for each participant with quality checks
participant_thresholds <- correctness %>%
  group_by(participant, hp_group, in_group) %>%
  do({
    # Check if participant has sufficient variability in responses
    response_var <- var(.$responses.user_response)
    n_trials <- nrow(.)

    # Only fit model if there's variation and sufficient data
    if(response_var > 0.01 && n_trials >= 10) {
      tryCatch({
        result <- calculate_threshold(.)

        # Flag extreme/invalid values
        if(abs(result$threshold_50) > 500 || abs(result$jnd) > 500) {
          data.frame(threshold_50 = NA, threshold_75 = NA, jnd = NA, slope = NA,
                     excluded = "extreme_values")
        } else {
          result$excluded <- "included"
          result
        }
      }, error = function(e) {
        data.frame(threshold_50 = NA, threshold_75 = NA, jnd = NA, slope = NA,
                   excluded = "model_error")
      })
    } else {
      data.frame(threshold_50 = NA, threshold_75 = NA, jnd = NA, slope = NA,
                 excluded = "insufficient_variability")
    }
  })
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
# Report exclusions
cat("\n=== DATA QUALITY CHECK ===\n")
```

```
##
## === DATA QUALITY CHECK ===
```

```r
exclusion_summary <- table(participant_thresholds$excluded)
cat("Participant exclusions:\n")
```

```
## Participant exclusions:
```

```r
print(exclusion_summary)
```

```
##
##           extreme_values                  included insufficient_variability
##                        3                        49                        1
```

```
# Filter to valid participants only
participant_thresholds_clean <- participant_thresholds %>%
  filter(!is.na(threshold_50))

cat(sprintf("\n%d participants included in analysis (%.1f%%)\n",
            nrow(participant_thresholds_clean),
            100 * nrow(participant_thresholds_clean) / length(unique(correctness$participant))))
```

```
##
## 49 participants included in analysis (92.5%)
```

```
# Check sample sizes
cat("Sample Sizes by Group:\n")
```

```
## Sample Sizes by Group:
```

```
print(table(participant_thresholds$hp_group, participant_thresholds$in_group))
```

```
##
##                      High Inattentive Low Inattentive
##   High Hyperactive                 16               9
##   Low Hyperactive                   9              19
```

```
# Descriptive statistics
cat("\n=== Descriptive Statistics ===\n\n")
```

```
##
## === Descriptive Statistics ===
```

```
cat("By Hyperactivity Group:\n")
```

```
## By Hyperactivity Group:
```

```
desc_hp <- participant_thresholds %>%
  group_by(hp_group) %>%
  summarise(
    Mean_Threshold = mean(threshold_50, na.rm = TRUE),
    SD_Threshold = sd(threshold_50, na.rm = TRUE),
    Mean_JND = mean(jnd, na.rm = TRUE),
    SD_JND = sd(jnd, na.rm = TRUE),
    Mean_Slope = mean(slope, na.rm = TRUE),
    N = n()
  )
print(desc_hp)
```

```
## # A tibble: 2 x 7
##   hp_group         Mean_Threshold SD_Threshold Mean_JND SD_JND Mean_Slope     N
##   <chr>                     <dbl>        <dbl>    <dbl>  <dbl>      <dbl> <int>
## 1 High Hyperactive           166.         124.     127.   121.     0.0311    25
## 2 Low Hyperactive            166.          88.5     85.7   82.8    0.109     28
```

```
cat("\nBy Inattention Group:\n")
```

```
##
## By Inattention Group:
```

```
desc_in <- participant_thresholds %>%
  group_by(in_group) %>%
  summarise(
    Mean_Threshold = mean(threshold_50, na.rm = TRUE),
    SD_Threshold = sd(threshold_50, na.rm = TRUE),
    Mean_JND = mean(jnd, na.rm = TRUE),
    SD_JND = sd(jnd, na.rm = TRUE),
    Mean_Slope = mean(slope, na.rm = TRUE),
    N = n()
  )
print(desc_in)
```

```
## # A tibble: 2 x 7
##   in_group         Mean_Threshold SD_Threshold Mean_JND SD_JND Mean_Slope     N
##   <chr>                     <dbl>        <dbl>    <dbl>  <dbl>      <dbl> <int>
## 1 High Inattentive           154.         106.     102.   95.4     0.0567    25
## 2 Low Inattentive            176.         103.     105.  110.      0.0923    28
```

```
# ANOVA for 50% threshold
cat("\n=== ANOVA: 50% Detection Threshold ===\n")
```

```
##
## === ANOVA: 50% Detection Threshold ===
```

```
anova_threshold <- aov(threshold_50 ~ hp_group * in_group, data = participant_thresholds)
print(summary(anova_threshold))
```

```
##                   Df Sum Sq Mean Sq F value Pr(>F)
## hp_group           1      3       3   0.000  0.987
## in_group           1   6870    6870   0.603  0.441
## hp_group:in_group  1    483     483   0.042  0.838
## Residuals         45 512454   11388
## 4 observations deleted due to missingness
```

```
# ANOVA for JND
cat("\n=== ANOVA: Just Noticeable Difference (JND) ===\n")
```

```
##
## === ANOVA: Just Noticeable Difference (JND) ===
```

```r
anova_jnd <- aov(jnd ~ hp_group * in_group, data = participant_thresholds)
print(summary(anova_jnd))
```

```
##                   Df Sum Sq Mean Sq F value Pr(>F)
## hp_group           1  20405   20405   1.948  0.170
## in_group           1   3777    3777   0.361  0.551
## hp_group:in_group  1   4747    4747   0.453  0.504
## Residuals         45 471401   10476
## 4 observations deleted due to missingness
```

```r
# ANOVA for slope (sensitivity)
cat("\n=== ANOVA: Psychometric Slope (Sensitivity) ===\n")
```

```
##
## === ANOVA: Psychometric Slope (Sensitivity) ===
```

```r
anova_slope <- aov(slope ~ hp_group * in_group, data = participant_thresholds)
print(summary(anova_slope))
```

```
##                   Df Sum Sq Mean Sq F value Pr(>F)
## hp_group           1 0.0727 0.07269   1.627  0.209
## in_group           1 0.0012 0.00117   0.026  0.872
## hp_group:in_group  1 0.0172 0.01719   0.385  0.538
## Residuals         45 2.0101 0.04467
## 4 observations deleted due to missingness
```

```r
# Post-hoc tests if significant
if(any(summary(anova_threshold)[[1]][["Pr(>F)"]][1:3] < 0.05, na.rm = TRUE)) {
  cat("\n=== Post-hoc: Threshold ===\n")
  cat("Hyperactivity:\n")
  print(pairwise.t.test(participant_thresholds$threshold_50,
                        participant_thresholds$hp_group,
                        p.adjust.method = "bonferroni"))
  cat("\nInattention:\n")
  print(pairwise.t.test(participant_thresholds$threshold_50,
                        participant_thresholds$in_group,
                        p.adjust.method = "bonferroni"))
}

if(any(summary(anova_jnd)[[1]][["Pr(>F)"]][1:3] < 0.05, na.rm = TRUE)) {
  cat("\n=== Post-hoc: JND ===\n")
  cat("Hyperactivity:\n")
  print(pairwise.t.test(participant_thresholds$jnd,
                        participant_thresholds$hp_group,
                        p.adjust.method = "bonferroni"))
  cat("\nInattention:\n")
  print(pairwise.t.test(participant_thresholds$jnd,
                        participant_thresholds$in_group,
                        p.adjust.method = "bonferroni"))
}

# Effect sizes (Cohen's d)
cat("\n=== Effect Sizes (Cohen's d) ===\n")
```

```
##
## === Effect Sizes (Cohen's d) ===
```

```
if(length(unique(participant_thresholds$hp_group)) == 2) {
  groups <- unique(participant_thresholds$hp_group)
  g1 <- participant_thresholds %>% filter(hp_group == groups[1])
  g2 <- participant_thresholds %>% filter(hp_group == groups[2])

  pooled_sd <- sqrt(((nrow(g1)-1)*sd(g1$jnd)^2 + (nrow(g2)-1)*sd(g2$jnd)^2) /
                    (nrow(g1) + nrow(g2) - 2))
  cohens_d <- (mean(g1$jnd) - mean(g2$jnd)) / pooled_sd

  cat(sprintf("\nHyperactivity JND: Cohen's d = %.3f\n", cohens_d))
  cat(sprintf("Interpretation: %s\n",
              ifelse(abs(cohens_d) < 0.2, "negligible",
                     ifelse(abs(cohens_d) < 0.5, "small",
                            ifelse(abs(cohens_d) < 0.8, "medium", "large")))))
}
```

```
##
## Hyperactivity JND: Cohen's d = NA
## Interpretation: NA
```

```
cat("\n=== SUMMARY ===\n")
```

```
##
## === SUMMARY ===
```

```
cat("Groups with HIGHER thresholds and JND have MORE difficulty\n")
```

```
## Groups with HIGHER thresholds and JND have MORE difficulty
```

```
cat("Groups with LOWER slopes have MORE difficulty (shallower curves)\n")
```

```
## Groups with LOWER slopes have MORE difficulty (shallower curves)
```

Data Quality: 9 out of 53 participants included (92.5%)

3 excluded for extreme values (those billion-millisecond outliers) 1 excluded for insufficient variability This is very good data quality!

Key Findings: STILL No Significant Effects Descriptive Statistics (Now Reasonable!): By Hyperactivity:

High Hyperactive: Threshold = 165.59ms, JND = 126.97ms Low Hyperactive: Threshold = 166.09ms, JND = 85.73ms

By Inattention:

High Inattentive: Threshold = 153.96ms, JND = 102.06ms Low Inattentive: Threshold = 176.42ms, JND = 104.59ms

ANOVA Results: 1. Detection Threshold: All p > 0.05 (NOT significant)

hp_group: F(1,45) = 0.000, p = 0.987 in_group: F(1,45) = 0.603, p = 0.441

2. JND (Difficulty measure): All p > 0.05 (NOT significant)

hp_group: F(1,45) = 1.948, p = 0.170 in_group: F(1,45) = 0.361, p = 0.551

3. Slope (Sensitivity): All p > 0.05 (NOT significant)

hp_group: F(1,45) = 1.627, p = 0.209

What This Means: For Hyperactivity:

High vs Low Hyperactive have nearly identical thresholds (~165-166ms) JND shows a trend: High Hyperactive = 127ms vs Low Hyperactive = 86ms

This is a 48% difference suggesting High Hyperactive have worse temporal discrimination But with p = 0.170, it's not statistically significant This could be a real effect that you're underpowered to detect

For Inattention:

Barely any difference in JND (102 vs 105ms) High Inattentive actually have slightly better thresholds (154 vs 176ms), opposite of expectation Clearly no effect here

#Power analysis

```r
# calculating Cohen's d properly
high_hp <- participant_thresholds_clean %>% filter(hp_group == "High Hyperactive")
low_hp <- participant_thresholds_clean %>% filter(hp_group == "Low Hyperactive")

cohens_d <- (mean(high_hp$jnd) - mean(low_hp$jnd)) /
            sqrt((sd(high_hp$jnd)^2 + sd(low_hp$jnd)^2) / 2)

# Power analysis with Cohen's d = 0.40
library(pwr)

pwr.t.test(d = 0.40,            # your observed effect size
           sig.level = 0.05,
           power = 0.80,
           type = "two.sample")
```

```
##
##      Two-sample t test power calculation
##
##               n = 99.08032
##               d = 0.4
##       sig.level = 0.05
##           power = 0.8
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

```r
#power we actually have
pwr.t.test(n = 26,              # average of 25 and 28
           d = 0.40,
           sig.level = 0.05,
           type = "two.sample")
```

```
##
##      Two-sample t test power calculation
##
##               n = 26
##               d = 0.4
##       sig.level = 0.05
##           power = 0.2931111
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

We would need 100 participants per group to detect the effect size observed with 80% power at = 0.05. We currently have 29% percent chance to detect this effect, therefore 71% chance of missing it even if it exists. We essentially flipped a coin that was 71% likely to come up “no significant difference”.

Cohen’s d = 0.40 - This is a SMALL TO MEDIUM Effect!What This Means: There IS a meaningful difference between groups:

- High Hyperactive participants have JND that’s about 0.4 standard deviations larger than Low Hyperactive
- This translates to the 48% difference we saw (127ms vs 86ms)
- This is not trivial - it’s a real, moderate effect

The problem is high variability + small sample size: standard deviations are huge,