Data analysis script 1: Create preliminary data for results from Study 1: State-Trait

Template Rmd

Brent Rappaport

2023-05-23

Contents

About

This script does preliminary data analysis comparing the psychometrics of HDDM models vs. raw accuracy vs. NIH Toolbox derived score for the Flanker task of the State-Trait study.

v = **drift rate**: "The parameter of primary interest for the present study is the drift rate, v, which is the average rate of approach to a boundary and indexes the quality or strength of evidence extracted from the stimulus. A large value of drift indicates strong decision evidence, meaning the decision process will approach the appropriate boundary quickly, leading to fast and accurate responses." Larger values of v mean faster accumulation of evidence (faster rate), larger t means slower non-decision time processing.

z = **response bias**: "If participants were biased toward one of the two responses (e.g., by increasing the proportion of one response over the other), they would move their starting point closer to that boundary. This produces faster and more probable responses at that boundary since less evidence is needed to reach it." Distance between the the start of the drift process and upper boundary.

a = **separation between the two boundaries**: "response caution or speed/accuracy tradeoffs. If the boundary separation is relatively small, responses will take less time to reach a boundary, leading to faster responses, but they will also be more likely to reach the wrong boundary due to noise in the process, leading to more errors." Larger a means more separation between the correct and incorrect response boundaries

t = **non-decision time**: "takes into account the duration of nondecisional processes. . . such processes may comprise basic encoding processes, the configuration of working memory for a task, and processes of response execution (i.e., motor activity)." (Voss et al., 2013)

Per Allie, larger values of v mean faster accumulation of evidence (faster rate), larger t means slower non-decision time processing, larger a means more separation between the correct and incorrect response boundaries, and z is the distance from the upper boundary to the start of the drift process.

Get Setup

Clear everything & set width

Load Libraries

```
library(knitr)      #allows rmarkdown files
library(haven)      #helps import stata
library(MASS)       #calculate residualized scores
library(tidyverse)  #plotting/cleaning, etc.
library(broom)      #nice statistical output
library(here)       #nice file paths
library(expss)      #labeling variables/values
library(psych)      #used for statistical analyses
library(labelled)
library(confintr)
library(papaja)
library(DescTools)
library(irr)
library(workflowr)  #helps with workflow
library(lmerTest)
```

```r
library(broom.mixed)
library(ggplot2)
```

Function correlation_matrix

```r
correlation_matrix <- function(df,
                               type = "spearman",
                               digits = 3,
                               decimal.mark = ".",
                               use = "all",
                               show_significance = TRUE,
                               replace_diagonal = FALSE,
                               replacement = ""){

  # check arguments
  stopifnot({
    is.numeric(digits)
    digits >= 0
    use %in% c("all", "upper", "lower")
    is.logical(replace_diagonal)
    is.logical(show_significance)
    is.character(replacement)
  })
  # we need the Hmisc package for this
  require(Hmisc)

  # retain only numeric and boolean columns
  isNumericOrBoolean = vapply(df, function(x) is.numeric(x) | is.logical(x), logical(1))
  if (sum(!isNumericOrBoolean) > 0) {
    cat('Dropping non-numeric/-boolean column(s):', paste(names(isNumericOrBoolean)[!isNumericOrBoolean]
  }
  df = df[isNumericOrBoolean]

  # transform input data frame to matrix
  x <- as.matrix(df)

  # run correlation analysis using Hmisc package
  correlation_matrix <- Hmisc::rcorr(x, type = )
  R <- correlation_matrix$r # Matrix of correlation coeficients
  p <- correlation_matrix$P # Matrix of p-value

  # transform correlations to specific character format
  Rformatted = formatC(R, format = 'f', digits = digits, decimal.mark = decimal.mark)

  # if there are any negative numbers, we want to put a space before the positives to align all
  if (sum(R < 0) > 0) {
    Rformatted = ifelse(R > 0, paste0(' ', Rformatted), Rformatted)
  }

  # add significance levels if desired
  if (show_significance) {
```

```
    # define notions for significance levels; spacing is important.
    stars <- ifelse(is.na(p), "   ", ifelse(p < .001, "***", ifelse(p < .01, "** ", ifelse(p < .05, "*
    Rformatted = paste0(Rformatted, stars)
  }
  # build a new matrix that includes the formatted correlations and their significance stars
  Rnew <- matrix(Rformatted, ncol = ncol(x))
  rownames(Rnew) <- colnames(x)
  colnames(Rnew) <- paste(colnames(x), "", sep =" ")

  # replace undesired values
  if (use == 'upper') {
    Rnew[lower.tri(Rnew, diag = replace_diagonal)] <- replacement
  } else if (use == 'lower') {
    Rnew[upper.tri(Rnew, diag = replace_diagonal)] <- replacement
  } else if (replace_diagonal) {
    diag(Rnew) <- replacement
  }

  return(Rnew)
}

save_correlation_matrix = function(df, filename, ...) {
  write.csv2(correlation_matrix(df, ...), file = filename)
}
```

Load Data

Remember to immediately rename and remove. Avoid overwriting old data.

```
here::i_am("work/analysis/do01_LDDM_sttr.Rmd")
```

```
## here() starts at /Users/brentrappaport/Documents/temp_files/DDM
```

```
# FULL DATA--ALL DAYS
## Load full multi-day dataset
load(file=here("work/data/LDDM_cleaning03.RData"))
LDDM_do1 <- LDDM_cleaning03; rm(LDDM_cleaning03)
## Load raw data across all days
load(file=here("work/data/LDDM_cleaning02_rt.RData"))
LDDM_do1_rt <- LDDM_cleaning02_rt; rm(LDDM_cleaning02_rt)

# DAY 1
## Load Day 1 dataset
load(file=here("work/data/LDDM_cleaning03_d1.RData"))
LDDM_do1_d1 <- LDDM_cleaning03_d1; rm(LDDM_cleaning03_d1)
# Load NIH Toolbox flanker score Day 1 dataset
load(file=here("work/data/LDDM_cleaning04_d1_calc4.RData"))
# Merge Day 1 dataset with NIH Toolbox flanker score Day 1 dataset
LDDM_do2_d1 <- left_join(LDDM_do1_d1, LDDM_cleaning04_d1_calc4, by="ID")
# Load Day 1 Alternative model results
LDDM_do_alt_d1 <- read.csv(here("DDM_Results/Block_Based_Day1/Day1_Block11_Alternative_Models.csv"), hea
LDDM_do2_alt_d1<- left_join(select(LDDM_do1_d1, contains("ID") | contains("ERN") | contains ("P3")), LDD
```

```
# DAY 2
## Load Day 2 dataset
load(file=here("work/data/LDDM_cleaning03_d2.RData"))
LDDM_do1_d2 <- LDDM_cleaning03_d2; rm(LDDM_cleaning03_d2)
# Load NIH Toolbox flanker score Day 2 dataset
load(file=here("work/data/LDDM_cleaning04_d2_calc4.RData"))
# Merge Day 2 dataset with NIH Toolbox flanker score Day 2 dataset
LDDM_do2_d2 <- left_join(LDDM_do1_d2, LDDM_cleaning04_d2_calc4, by="ID")

# DAY 3
## Load Day 3 dataset
load(file=here("work/data/LDDM_cleaning03_d3.RData"))
LDDM_do1_d3 <- LDDM_cleaning03_d3; rm(LDDM_cleaning03_d3)
# Load NIH Toolbox flanker score Day 3 dataset
load(file=here("work/data/LDDM_cleaning04_d3_calc4.RData"))
# Merge Day 3 dataset with NIH Toolbox flanker score Day 3 dataset
LDDM_do2_d3 <- left_join(LDDM_do1_d3, LDDM_cleaning04_d3_calc4, by="ID")
```

Winsorize outliers

```
LDDM_do2_d1_not_outliers <- LDDM_do2_d1 %>%
  mutate(FCZ_ERN_080 = Winsorize(FCZ_ERN_080,
                                 minval=mean(FCZ_ERN_080, na.rm=T)-(3*sd(FCZ_ERN_080, na.rm=T)),
                                 maxval=mean(FCZ_ERN_080, na.rm=T)+(3*sd(FCZ_ERN_080, na.rm=T)),
                                 na.rm=T)) %>%
  mutate(FCZ_ERN_080 = FCZ_ERN_080*-1)

LDDM_do2_d2_not_outliers <- LDDM_do2_d2 %>%
  mutate(FCZ_ERN_080 = Winsorize(FCZ_ERN_080,
                                 minval=mean(FCZ_ERN_080, na.rm=T)-(3*sd(FCZ_ERN_080, na.rm=T)),
                                 maxval=mean(FCZ_ERN_080, na.rm=T)+(3*sd(FCZ_ERN_080, na.rm=T)),
                                 na.rm=T)) %>%
  mutate(FCZ_ERN_080 = FCZ_ERN_080*-1)

LDDM_do2_d3_not_outliers <- LDDM_do2_d3 %>%
  mutate(FCZ_ERN_080 = Winsorize(FCZ_ERN_080,
                                 minval=mean(FCZ_ERN_080, na.rm=T)-(3*sd(FCZ_ERN_080, na.rm=T)),
                                 maxval=mean(FCZ_ERN_080, na.rm=T)+(3*sd(FCZ_ERN_080, na.rm=T)),
                                 na.rm=T)) %>%
  mutate(FCZ_ERN_080 = FCZ_ERN_080*-1)
```

Standardize scores

```
LDDM_do2_d1$v_S1_B11 <- rowMeans(LDDM_do2_d1[,c('v_congruent_S1_B11','v_incongruent_S1_B11')], na.rm=F)
LDDM_do2_d1_not_outliers$v_S1_B11 <- rowMeans(LDDM_do2_d1_not_outliers[,c('v_congruent_S1_B11','v_incong
LDDM_do2_d2$v_S2_B11 <- rowMeans(LDDM_do2_d2[,c('v_congruent_S2_B11','v_incongruent_S2_B11')], na.rm=F)
LDDM_do2_d2_not_outliers$v_S2_B11 <- rowMeans(LDDM_do2_d2_not_outliers[,c('v_congruent_S2_B11','v_incong
LDDM_do2_d3$v_S3_B11 <- rowMeans(LDDM_do2_d3[,c('v_congruent_S3_B11','v_incongruent_S3_B11')], na.rm=F)
```

```
LDDM_do2_d3_not_outliers$v_S3_B11 <- rowMeans(LDDM_do2_d3_not_outliers[,c('v_congruent_S3_B11','v_incong

var_list_d1 <- c("v_S1_B11","v_congruent_S1_B11","v_incongruent_S1_B11","a_S1_B11","t_S1_B11","z_S1_B11
              "flanker_score","accuracy","accuracy_congruent_log","accuracy_incongruent",
              "FCZ_ERN_080","PZ_P3_onset_con","PZ_P3_onset_incon")

var_list_d2 <- c("v_S2_B11","v_congruent_S2_B11","v_incongruent_S2_B11","a_S2_B11","t_S2_B11","z_S2_B11
              "flanker_score","accuracy","accuracy_congruent_log","accuracy_incongruent",
              "FCZ_ERN_080","PZ_P3_onset_con","PZ_P3_onset_incon")

var_list_d3 <- c("v_S3_B11","v_congruent_S3_B11","v_incongruent_S3_B11","a_S3_B11","t_S3_B11","z_S3_B11
              "flanker_score","accuracy","accuracy_congruent_log","accuracy_incongruent",
              "FCZ_ERN_080","PZ_P3_onset_con","PZ_P3_onset_incon")

var_list_alt <- c("B11_avt_v","B11_avt_a","B11_avt_t",
              "B11_avtz_v","B11_avtz_a","B11_avtz_t","B11_avtz_z",
              "B11_avtz_DO_vt_v_con","B11_avtz_DO_vt_v_incon","B11_avtz_DO_vt_a", "B11_avtz_DO_vt_t_con
              "B11_avtz_DO_vtz_v_con","B11_avtz_DO_vtz_v_incon","B11_avtz_DO_vtz_a","B11_avtz_DO_vtz_t_
              "FCZ_ERN_080","PZ_P3_onset_con","PZ_P3_onset_incon")

for (v in var_list_d1){
  print(paste0("LDDM_do2_d1$",v))
  eval(parse(text=paste0('LDDM_do2_d1$',v,'_d1_z <- scale(LDDM_do2_d1$',v,', center=T, scale=T)')))
  eval(parse(text=paste0('LDDM_do2_d1_not_outliers$',v,'_d1_z <- scale(LDDM_do2_d1_not_outliers$',v,', c
}
```

```
## [1] "LDDM_do2_d1$v_S1_B11"
## [1] "LDDM_do2_d1$v_congruent_S1_B11"
## [1] "LDDM_do2_d1$v_incongruent_S1_B11"
## [1] "LDDM_do2_d1$a_S1_B11"
## [1] "LDDM_do2_d1$t_S1_B11"
## [1] "LDDM_do2_d1$z_S1_B11"
## [1] "LDDM_do2_d1$flanker_score"
## [1] "LDDM_do2_d1$accuracy"
## [1] "LDDM_do2_d1$accuracy_congruent_log"
## [1] "LDDM_do2_d1$accuracy_incongruent"
## [1] "LDDM_do2_d1$FCZ_ERN_080"
## [1] "LDDM_do2_d1$PZ_P3_onset_con"
## [1] "LDDM_do2_d1$PZ_P3_onset_incon"
```

```
for (v in var_list_alt){
  print(paste0("LDDM_do2_alt_d1$",v))
  eval(parse(text=paste0('LDDM_do2_alt_d1$',v,'_d1_z <- scale(LDDM_do2_alt_d1$',v,', center=T, scale=T)
}
```

```
## [1] "LDDM_do2_alt_d1$B11_avt_v"
## [1] "LDDM_do2_alt_d1$B11_avt_a"
## [1] "LDDM_do2_alt_d1$B11_avt_t"
## [1] "LDDM_do2_alt_d1$B11_avtz_v"
## [1] "LDDM_do2_alt_d1$B11_avtz_a"
## [1] "LDDM_do2_alt_d1$B11_avtz_t"
## [1] "LDDM_do2_alt_d1$B11_avtz_z"
## [1] "LDDM_do2_alt_d1$B11_avtz_DO_vt_v_con"
```

```
## [1] "LDDM_do2_alt_d1$B11_avtz_DO_vt_v_incon"
## [1] "LDDM_do2_alt_d1$B11_avtz_DO_vt_a"
## [1] "LDDM_do2_alt_d1$B11_avtz_DO_vt_t_con"
## [1] "LDDM_do2_alt_d1$B11_avtz_DO_vt_t_incon"
## [1] "LDDM_do2_alt_d1$B11_avtz_DO_vt_z"
## [1] "LDDM_do2_alt_d1$B11_avtz_DO_vtz_v_con"
## [1] "LDDM_do2_alt_d1$B11_avtz_DO_vtz_v_incon"
## [1] "LDDM_do2_alt_d1$B11_avtz_DO_vtz_a"
## [1] "LDDM_do2_alt_d1$B11_avtz_DO_vtz_t_con"
## [1] "LDDM_do2_alt_d1$B11_avtz_DO_vtz_t_incon"
## [1] "LDDM_do2_alt_d1$B11_avtz_DO_vtz_z_con"
## [1] "LDDM_do2_alt_d1$B11_avtz_DO_vtz_z_incon"
## [1] "LDDM_do2_alt_d1$FCZ_ERN_080"
## [1] "LDDM_do2_alt_d1$PZ_P3_onset_con"
## [1] "LDDM_do2_alt_d1$PZ_P3_onset_incon"
```

```
for (v in var_list_d2){
  print(paste0("LDDM_do2_d2$",v))
  eval(parse(text=paste0('LDDM_do2_d2$',v,'_d2_z <- scale(LDDM_do2_d2$',v,', center=T, scale=T)')))
  eval(parse(text=paste0('LDDM_do2_d2_not_outliers$',v,'_d2_z <- scale(LDDM_do2_d2_not_outliers$',v,', 
}
```

```
## [1] "LDDM_do2_d2$v_S2_B11"
## [1] "LDDM_do2_d2$v_congruent_S2_B11"
## [1] "LDDM_do2_d2$v_incongruent_S2_B11"
## [1] "LDDM_do2_d2$a_S2_B11"
## [1] "LDDM_do2_d2$t_S2_B11"
## [1] "LDDM_do2_d2$z_S2_B11"
## [1] "LDDM_do2_d2$flanker_score"
## [1] "LDDM_do2_d2$accuracy"
## [1] "LDDM_do2_d2$accuracy_congruent_log"
## [1] "LDDM_do2_d2$accuracy_incongruent"
## [1] "LDDM_do2_d2$FCZ_ERN_080"
## [1] "LDDM_do2_d2$PZ_P3_onset_con"
## [1] "LDDM_do2_d2$PZ_P3_onset_incon"
```

```
for (v in var_list_d3){
  print(paste0("LDDM_do2_d3$",v))
  eval(parse(text=paste0('LDDM_do2_d3$',v,'_d3_z <- scale(LDDM_do2_d3$',v,', center=T, scale=T)')))
  eval(parse(text=paste0('LDDM_do2_d3_not_outliers$',v,'_d3_z <- scale(LDDM_do2_d3_not_outliers$',v,', 
}
```

```
## [1] "LDDM_do2_d3$v_S3_B11"
## [1] "LDDM_do2_d3$v_congruent_S3_B11"
## [1] "LDDM_do2_d3$v_incongruent_S3_B11"
## [1] "LDDM_do2_d3$a_S3_B11"
## [1] "LDDM_do2_d3$t_S3_B11"
## [1] "LDDM_do2_d3$z_S3_B11"
## [1] "LDDM_do2_d3$flanker_score"
## [1] "LDDM_do2_d3$accuracy"
## [1] "LDDM_do2_d3$accuracy_congruent_log"
## [1] "LDDM_do2_d3$accuracy_incongruent"
## [1] "LDDM_do2_d3$FCZ_ERN_080"
```

```
## [1] "LDDM_do2_d3$PZ_P3_onset_con"
## [1] "LDDM_do2_d3$PZ_P3_onset_incon"
```

Correlations

ERN amplitude & DDM

```
all_measures_d1 <- c("FCZ_ERN_080_d1_z","flanker_score_d1_z","accuracy_d1_z","accuracy_incongruent_d1_z
                     "v_S1_B11_d1_z","v_congruent_S1_B11_d1_z","v_incongruent_S1_B11_d1_z","a_S1_B11_d1_z"
all_measures_d2 <- c("FCZ_ERN_080_d2_z","flanker_score_d2_z","accuracy_d2_z","accuracy_incongruent_d2_z
                     "v_S2_B11_d2_z","v_congruent_S2_B11_d2_z","v_incongruent_S2_B11_d2_z","a_S2_B11_d2_z"
all_measures_d3 <- c("FCZ_ERN_080_d3_z","flanker_score_d3_z","accuracy_d3_z","accuracy_incongruent_d3_z
                     "v_S3_B11_d3_z","v_congruent_S3_B11_d3_z","v_incongruent_S3_B11_d3_z","a_S3_B11_d3_z"


sttr_d1_correlations <- correlation_matrix(LDDM_do2_d1_not_outliers[c(all_measures_d1)], type = c("spea
```

```
## Loading required package: Hmisc
```

```
## Registered S3 methods overwritten by 'Hmisc':
##   method                 from
##   [.labelled             expss
##   print.labelled         expss
##   as.data.frame.labelled expss
```

```
##
## Attaching package: 'Hmisc'
```

```
## The following objects are masked from 'package:DescTools':
##
##     %nin%, Label, Mean, Quantile
```

```
## The following object is masked from 'package:psych':
##
##     describe
```

```
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
sttr_d2_correlations <- correlation_matrix(LDDM_do2_d2_not_outliers[c(all_measures_d2)], type = c("spea
sttr_d3_correlations <- correlation_matrix(LDDM_do2_d3_not_outliers[c(all_measures_d3)], type = c("spea

LDDM_do2_not_outliers <- full_join(LDDM_do2_d1_not_outliers, LDDM_do2_d2_not_outliers, by="ID") %>%
  full_join(LDDM_do2_d3_not_outliers, by="ID")
```

```
sttr_correlations <- correlation_matrix(LDDM_do2_not_outliers[c(all_measures_d1,all_measures_d2,all_meas

write.csv(sttr_d1_correlations, file=here("work/tables/sttr_d1_correlations.csv"))
write.csv(sttr_d2_correlations, file=here("work/tables/sttr_d2_correlations.csv"))
write.csv(sttr_d3_correlations, file=here("work/tables/sttr_d3_correlations.csv"))
write.csv(sttr_correlations, file=here("work/tables/sttr_correlations.csv"))

# ERN_table_d1 <- data.frame(parameters= c("v", "v_congruent", "v_incongruent", "a", "t", "z"),
#                            r= c(corr.test(LDDM_do2_d1_not_outliers$FCZ_ERN_080_d1_z, LDDM_do2_d1_not_out
#                                 corr.test(LDDM_do2_d1_not_outliers$FCZ_ERN_080_d1_z, LDDM_do2_d1_not_out
#                                 corr.test(LDDM_do2_d1_not_outliers$FCZ_ERN_080_d1_z, LDDM_do2_d1_not_out
#                                 corr.test(LDDM_do2_d1_not_outliers$FCZ_ERN_080_d1_z, LDDM_do2_d1_not_out
#                                 corr.test(LDDM_do2_d1_not_outliers$FCZ_ERN_080_d1_z, LDDM_do2_d1_not_out
#                                 corr.test(LDDM_do2_d1_not_outliers$FCZ_ERN_080_d1_z, LDDM_do2_d1_not_out
#                            p_nominal = c(corr.test(LDDM_do2_d1_not_outliers$FCZ_ERN_080_d1_z, LDDM_do2_d
#                                 corr.test(LDDM_do2_d1_not_outliers$FCZ_ERN_080_d1_z, LDDM_do2_d1_not_out
#                                 corr.test(LDDM_do2_d1_not_outliers$FCZ_ERN_080_d1_z, LDDM_do2_d1_not_out
#                                 corr.test(LDDM_do2_d1_not_outliers$FCZ_ERN_080_d1_z, LDDM_do2_d1_not_out
#                                 corr.test(LDDM_do2_d1_not_outliers$FCZ_ERN_080_d1_z, LDDM_do2_d1_not_out
#                                 corr.test(LDDM_do2_d1_not_outliers$FCZ_ERN_080_d1_z, LDDM_do2_d1_not_out
# ERN_table_d1

# write.csv(ERN_table_d1, here("work/tables/sttr_ERN_table_d1.csv"))

# ggplot(LDDM_do2_d1, aes(x=FCZ_ERN_080_d1_z, y=v_S1_B11)) +
#   geom_point() +
#   stat_smooth(method="lm")
```

Alternative models

```
ERN_table_d1_alt1 <- data.frame(parameters= c("v", "a", "t"),
                          r= c(corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avt_v_d1_z
                               corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avt_a_d1_z
                               corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avt_t_d1_z
                          p_nominal = c(corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_a
                               corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avt_a_d1_z
                               corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avt_t_d1_z

ERN_table_d1_alt2 <- data.frame(parameters= c("v", "a", "t", "z"),
                          r= c(corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_v_d1_
                               corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_a_d1_
                               corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_t_d1_
                               corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_z_d1_
                          p_nominal = c(corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_a
                               corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_a_d1_
                               corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_t_d1_
                               corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_z_d1_

ERN_table_d1_alt3 <- data.frame(parameters= c("v_congruent", "v_incongruent", "a", "t_congruent","t_inc
                          r= c(corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                               corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
```

```
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                         p_nominal = c(corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_av
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt

ERN_table_d1_alt4 <- data.frame(parameters= c("v_congruent", "v_incongruent", "a", "t_congruent","t_inc
                         r= c(corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                         p_nominal = c(corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_av
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt
                              corr.test(LDDM_do2_alt_d1$FCZ_ERN_080_d1_z, LDDM_do2_alt_d1$B11_avtz_DO_vt

write.csv(t(ERN_table_d1_alt1), here("work/tables/ERN_table_d1_alt1.csv"))
write.csv(t(ERN_table_d1_alt2), here("work/tables/ERN_table_d1_alt2.csv"))
write.csv(t(ERN_table_d1_alt3), here("work/tables/ERN_table_d1_alt3.csv"))
write.csv(t(ERN_table_d1_alt4), here("work/tables/ERN_table_d1_alt4.csv"))
```

Multiple regressions

Day 1

```
make_CI <- function(lower, upper) {
  paste0("[", round(lower,2), ", ", round(upper,2), "]")
}

i=1
mr_ERN_table_d1 <- as.data.frame(matrix(nrow=6, ncol=4))
mr_ERN_table_d1[,1] <- c("Drift rate","Drift rate (congruent)","Drift rate (incongruent)","Boundary sepa
mr_ERN_table_d1_raw <- as.data.frame(matrix(nrow=6, ncol=4))
mr_ERN_table_d1_raw[,1] <- c("Drift rate","Drift rate (congruent)","Drift rate (incongruent)","Boundary

for (var in c("v_S1_B11_d1_z","v_congruent_S1_B11_d1_z","v_incongruent_S1_B11_d1_z","a_S1_B11_d1_z","t_S
  eval(parse(text=paste0("
           model <- lm(FCZ_ERN_080_d1_z ~ ",var," + flanker_score_d1_z + accuracy_incongruent_d1_z, LD

           mr_ERN_table_d1[i,2] <- paste0(round(tidy(model)$estimate[2],2), ifelse(tidy(model)$p.valu
```

```
                                        make_CI(confint(model)[2,1], confint(model)[2,2]))
            mr_ERN_table_d1[i,3] <- paste0(round(tidy(model)$estimate[3],2), ifelse(tidy(model)$p.value
                                        make_CI(confint(model)[3,1], confint(model)[3,2]))
            mr_ERN_table_d1[i,4] <- paste0(round(tidy(model)$estimate[4],2), ifelse(tidy(model)$p.value
                                        make_CI(confint(model)[4,1], confint(model)[4,2]))

           mr_ERN_table_d1_raw[i,2] <- tidy(model)$estimate[2]
           mr_ERN_table_d1_raw[i,3] <- confint(model)[2,1]
           mr_ERN_table_d1_raw[i,4] <- confint(model)[2,2]

           mr_ERN_table_d1_raw[i,5] <- tidy(model)$estimate[3]
           mr_ERN_table_d1_raw[i,6] <- confint(model)[3,1]
           mr_ERN_table_d1_raw[i,7] <- confint(model)[3,2]

           mr_ERN_table_d1_raw[i,8] <- tidy(model)$estimate[4]
           mr_ERN_table_d1_raw[i,9] <- confint(model)[4,1]
           mr_ERN_table_d1_raw[i,10] <- confint(model)[4,2]
                                "")))
  i=i+1
}
colnames(mr_ERN_table_d1) <- c("Parameters","DDM", "NIH Toolbox", "Raw accuracy")

write.csv(mr_ERN_table_d1, here("work/tables/mr_ERN_table_d1.csv"))
write.csv(mr_ERN_table_d1_raw, here("work/tables/mr_ERN_table_d1_raw.csv"))

# ggplot(filter(LDDM_do2_d1_not_outliers), aes(x=FCZ_ERN_080_d1_z, y=flanker_score_d1)) +
#   geom_point() +
#   stat_smooth(method="lm")
#
# ggplot(filter(LDDM_do2_d1_not_outliers, flanker_score_d1>4), aes(x=FCZ_ERN_080_d1_z, y=flanker_score_
#   geom_point() +
#   stat_smooth(method="lm")
```

Day 2

```
i=1
mr_ERN_table_d2 <- as.data.frame(matrix(nrow=6, ncol=4))
mr_ERN_table_d2[,1] <- c("Drift rate","Drift rate (congruent)","Drift rate (incongruent)","Boundary sepa
mr_ERN_table_d2_raw <- as.data.frame(matrix(nrow=6, ncol=4))
mr_ERN_table_d2_raw[,1] <- c("Drift rate","Drift rate (congruent)","Drift rate (incongruent)","Boundary

for (var in c("v_S2_B11_d2_z","v_congruent_S2_B11_d2_z","v_incongruent_S2_B11_d2_z","a_S2_B11_d2_z","t_S
  eval(parse(text=paste0("
            model <- lm(FCZ_ERN_080_d2_z ~ ",var," + flanker_score_d2_z + accuracy_incongruent_d2_z, LL

            mr_ERN_table_d2[i,2] <- paste0(round(tidy(model)$estimate[2],2), ifelse(tidy(model)$p.value
                                        make_CI(confint(model)[2,1], confint(model)[2,2]))
            mr_ERN_table_d2[i,3] <- paste0(round(tidy(model)$estimate[3],2), ifelse(tidy(model)$p.value
                                        make_CI(confint(model)[3,1], confint(model)[3,2]))
            mr_ERN_table_d2[i,4] <- paste0(round(tidy(model)$estimate[4],2), ifelse(tidy(model)$p.value
                                        make_CI(confint(model)[4,1], confint(model)[4,2]))
```

```
            mr_ERN_table_d2_raw[i,2] <- tidy(model)$estimate[2]
            mr_ERN_table_d2_raw[i,3] <- confint(model)[2,1]
            mr_ERN_table_d2_raw[i,4] <- confint(model)[2,2]

            mr_ERN_table_d2_raw[i,5] <- tidy(model)$estimate[3]
            mr_ERN_table_d2_raw[i,6] <- confint(model)[3,1]
            mr_ERN_table_d2_raw[i,7] <- confint(model)[3,2]

            mr_ERN_table_d2_raw[i,8] <- tidy(model)$estimate[4]
            mr_ERN_table_d2_raw[i,9] <- confint(model)[4,1]
            mr_ERN_table_d2_raw[i,10] <- confint(model)[4,2]
                                "")))
  i=i+1
}
colnames(mr_ERN_table_d2) <- c("Parameters","DDM", "NIH Toolbox", "Raw accuracy")

write.csv(mr_ERN_table_d2, here("work/tables/mr_ERN_table_d2.csv"))
write.csv(mr_ERN_table_d2_raw, here("work/tables/mr_ERN_table_d2_raw.csv"))

# ggplot(filter(LDDM_do2_d2), aes(x=FCZ_ERN_080_d1_z, y=flanker_score_d2)) +
#   geom_point() +
#   stat_smooth(method="lm")
#
# ggplot(filter(LDDM_do2_d2, flanker_score_d2>4), aes(x=FCZ_ERN_080_d1_z, y=flanker_score_d2)) +
#   geom_point() +
#   stat_smooth(method="lm")
#
# ggplot(filter(LDDM_do2_d2, flanker_score_d2>4), aes(x=FCZ_ERN_080_d1_z, y=v_S2_B11)) +
#   geom_point() +
#   stat_smooth(method="lm")
```

Day 3

```
i=1
mr_ERN_table_d3 <- as.data.frame(matrix(nrow=6, ncol=4))
mr_ERN_table_d3[,1] <- c("Drift rate","Drift rate (congruent)","Drift rate (incongruent)","Boundary sepa
mr_ERN_table_d3_raw <- as.data.frame(matrix(nrow=6, ncol=4))
mr_ERN_table_d3_raw[,1] <- c("Drift rate","Drift rate (congruent)","Drift rate (incongruent)","Boundary

for (var in c("v_S3_B11_d3_z","v_congruent_S3_B11_d3_z","v_incongruent_S3_B11_d3_z","a_S3_B11_d3_z","t_S
  eval(parse(text=paste0("
            model <- lm(FCZ_ERN_080_d3_z ~ ",var," + flanker_score_d3_z + accuracy_incongruent_d3_z, LL

            mr_ERN_table_d3[i,2] <- paste0(round(tidy(model)$estimate[2],2), ifelse(tidy(model)$p.value
                                    make_CI(confint(model)[2,1], confint(model)[2,2]))
            mr_ERN_table_d3[i,3] <- paste0(round(tidy(model)$estimate[3],2), ifelse(tidy(model)$p.value
                                    make_CI(confint(model)[3,1], confint(model)[3,2]))
            mr_ERN_table_d3[i,4] <- paste0(round(tidy(model)$estimate[4],2), ifelse(tidy(model)$p.value
                                    make_CI(confint(model)[4,1], confint(model)[4,2]))

           mr_ERN_table_d3_raw[i,2] <- tidy(model)$estimate[2]
           mr_ERN_table_d3_raw[i,3] <- confint(model)[2,1]
```

```
            mr_ERN_table_d3_raw[i,4] <- confint(model)[2,2]

            mr_ERN_table_d3_raw[i,5] <- tidy(model)$estimate[3]
            mr_ERN_table_d3_raw[i,6] <- confint(model)[3,1]
            mr_ERN_table_d3_raw[i,7] <- confint(model)[3,2]

            mr_ERN_table_d3_raw[i,8] <- tidy(model)$estimate[4]
            mr_ERN_table_d3_raw[i,9] <- confint(model)[4,1]
            mr_ERN_table_d3_raw[i,10] <- confint(model)[4,2]
                                "")))
  i=i+1
}
colnames(mr_ERN_table_d3) <- c("Parameters","DDM", "NIH Toolbox", "Raw accuracy")

write.csv(mr_ERN_table_d3, here("work/tables/mr_ERN_table_d3.csv"))
write.csv(mr_ERN_table_d3_raw, here("work/tables/mr_ERN_table_d3_raw.csv"))

# ggplot(filter(LDDM_do2_d3), aes(x=FCZ_ERN_080_d1_z, y=flanker_score_d3)) +
#   geom_point() +
#   stat_smooth(method="lm")
#
# ggplot(filter(LDDM_do2_d3, flanker_score_d3>4), aes(x=FCZ_ERN_080_d1_z, y=flanker_score_d3)) +
#   geom_point() +
#   stat_smooth(method="lm")
#
# ggplot(filter(LDDM_do2_d3, flanker_score_d3>4), aes(x=FCZ_ERN_080_d1_z, y=v_S3_B11)) +
#   geom_point() +
#   stat_smooth(method="lm")
```

Reliability

Test-retest (ICC)

```
LDDM_do2_irr <- full_join(LDDM_do2_d1, LDDM_do2_d2, by="ID") %>%
  full_join(LDDM_do2_d3, by="ID")

library("irr")

# sttr_icc_table_12 <- data.frame(parameters= c("v", "v_congruent", "v_incongruent", "a", "t", "z", "ni
#                             ICC=c(icc(LDDM_do2_irr[c("v_S1_B11","v_S2_B11")])$value,
#                             icc(LDDM_do2_irr[c("v_congruent_S1_B11.x","v_congruent_S2_B11.x")])$value,
#                             icc(LDDM_do2_irr[c("v_incongruent_S1_B11.x","v_incongruent_S2_B11.x")])$va
#                             icc(LDDM_do2_irr[c("a_S1_B11.x","a_S2_B11.x")])$value,
#                             icc(LDDM_do2_irr[c("t_S1_B11.x","t_S2_B11.x")])$value,
#                             icc(LDDM_do2_irr[c("z_S1_B11.x","z_S2_B11.x")])$value,
#                             icc(LDDM_do2_irr[c("flanker_score_d1","flanker_score_d2")])$value))
#
# sttr_icc_table_23 <- data.frame(parameters= c("v", "v_congruent", "v_incongruent", "a", "t", "z", "ni
#                             ICC=c(icc(LDDM_do2_irr[c("v_S1_B11","v_S2_B11","v_S3_B11")])$value,
#                             icc(LDDM_do2_irr[c("v_congruent_S2_B11.x","v_congruent_S3_B11.x")])$value,
```

```
#                             icc(LDDM_do2_irr[c("v_incongruent_S2_B11.x","v_incongruent_S3_B11.x")])$va
#                             icc(LDDM_do2_irr[c("a_S2_B11.x","a_S3_B11.x")])$value,
#                             icc(LDDM_do2_irr[c("t_S2_B11.x","t_S3_B11.x")])$value,
#                             icc(LDDM_do2_irr[c("z_S2_B11.x","z_S3_B11.x")])$value,
#                             icc(LDDM_do2_irr[c("flanker_score_d2","flanker_score_d3")])$value))

sttr_icc_table_123 <- data.frame(parameters= c("Drift rate", "Drift rate (congruent)", "Drift rate (inco
                           ICC=c(icc(LDDM_do2_irr[c("v_S1_B11","v_S2_B11","v_S3_B11")])$value,
                           icc(LDDM_do2_irr[c("v_congruent_S1_B11.x","v_congruent_S2_B11.x","v_congruent
                           icc(LDDM_do2_irr[c("v_incongruent_S1_B11.x","v_incongruent_S2_B11.x","v_incon
                           icc(LDDM_do2_irr[c("a_S1_B11.x","a_S2_B11.x","a_S3_B11.x")])$value,
                           icc(LDDM_do2_irr[c("t_S1_B11.x","t_S2_B11.x","t_S3_B11.x")])$value,
                           icc(LDDM_do2_irr[c("z_S1_B11.x","z_S2_B11.x","z_S3_B11.x")])$value,
                           icc(LDDM_do2_irr[c("flanker_score_d1_z","flanker_score_d2_z","flanker_score_
                           icc(LDDM_do2_irr[c("accuracy_d1_z","accuracy_d2_z","accuracy_d3_z")])$value,
                           icc(LDDM_do2_irr[c("accuracy_incongruent_d1_z","accuracy_incongruent_d2_z","a

# write.csv(sttr_icc_table_12, "work/tables/sttr_icc_table_12.csv")
# write.csv(sttr_icc_table_23, "work/tables/sttr_icc_table_23.csv")
write.csv(sttr_icc_table_123, here("work/tables/sttr_icc_table_123.csv"))
```

Split half reliability (Spearman-Brown prophecy)

Day 1

```
load(file=here("work/data/LDDM_cleaning04_fullbeh_d1.RData"))
library(confintr)

LDDM_cleaning04_d1_reliability <- LDDM_cleaning04_fullbeh_d1 %>%
  group_by(subj_idx) %>%
  mutate(block = c(rep(1,30),rep(2,30),rep(3,30),rep(4,30),rep(5,30),rep(6,30),rep(7,30),rep(8,30),rep(
  group_by(subj_idx) %>%
  mutate(trial = 1:330)

for (s in seq(15,166,15)){
    eval(parse(text=paste0('
LDDM_first',s,'_1 <- LDDM_cleaning04_d1_reliability %>% filter(trial<=',s,')
LDDM_second',s,'_1 <- LDDM_cleaning04_d1_reliability %>% filter(trial< ',(s*2)+1,' & trial>',s,')

LDDM_first',s,'_2 <- LDDM_first',s,'_1 %>%
  group_by(subj_idx) %>% # per subject
  summarise(accuracy_score = sum(response==1)*(5/length(trial))) # accuracy score per NIH TOoolbox manua
LDDM_second',s,'_2 <- LDDM_second',s,'_1 %>%
  group_by(subj_idx) %>%
  summarise(accuracy_score = sum(response==1)*(5/length(trial))) # accuracy score per NIH TOoolbox manua

LDDM_first',s,'_3 <-  LDDM_first',s,'_1 %>%
  group_by(subj_idx) %>% # per subject
  filter(stim=="incongruent" & response==1) %>% # incongruent trials with correct response
  mutate(mean_rt = mean(rt),
         sd_rt = sd(rt)) %>% # compute individual mean and sd RT for use below
```

```
  filter(rt>=0.1 & rt>(mean_rt - 3*sd_rt) & rt<(mean_rt + 3*sd_rt)) %>% # remove trials less than 100ms
  summarise(med_rt = median(rt)*1000) %>% # compute individual level median RT
  mutate(rt_score = 5-(5*((log(med_rt)-log(250))/(log(1000)-log(250))))) # compute RT score to go into
LDDM_second',s,'_3 <-  LDDM_second',s,'_1 %>%
  group_by(subj_idx) %>% # per subject
  filter(stim=="incongruent" & response==1) %>% # incongruent trials with correct response
  mutate(mean_rt = mean(rt),
         sd_rt = sd(rt)) %>% # compute individual mean and sd RT for use below
  filter(rt>=0.1 & rt>(mean_rt - 3*sd_rt) & rt<(mean_rt + 3*sd_rt)) %>% # remove trials less than 100ms
  summarise(med_rt = median(rt)*1000) %>% # compute individual level median RT
  mutate(rt_score = 5-(5*((log(med_rt)-log(250))/(log(1000)-log(250))))) # compute RT score to go into

LDDM_first',s,' <- LDDM_first',s,'_1 %>%
  full_join(LDDM_first',s,'_2, by = "subj_idx") %>%
  full_join(LDDM_first',s,'_3, by="subj_idx") %>%
  group_by(subj_idx) %>% # per subject
  mutate(total_accuracy_perc = sum(response==1)/length(response)) %>% # compute total accuracy percentag
  summarise(flanker_score_list_d1 = if_else(total_accuracy_perc>=0.8, accuracy_score+rt_score, accuracy
            accuracy = mean(total_accuracy_perc)) %>% # if accuracy is above 80% then add accuracy and
  transmute(ID=subj_idx, flanker_score_list_d1=flanker_score_list_d1, accuracy=accuracy) %>%
  group_by(ID) %>% # per subject
  summarise(flanker_score_d1 = mean(flanker_score_list_d1),
            accuracy = mean(accuracy))
LDDM_second',s,' <- LDDM_second',s,'_1 %>%
  full_join(LDDM_second',s,'_2, by = "subj_idx") %>%
  full_join(LDDM_second',s,'_3, by="subj_idx") %>%
  group_by(subj_idx) %>% # per subject
  mutate(total_accuracy_perc = sum(response==1)/length(response)) %>% # compute total accuracy percentag
  summarise(flanker_score_list_d1 = if_else(total_accuracy_perc>=0.8, accuracy_score+rt_score, accuracy
            accuracy = mean(total_accuracy_perc)) %>% # if accuracy is above 80% then add accuracy and
  transmute(ID=subj_idx, flanker_score_list_d1=flanker_score_list_d1, accuracy=accuracy) %>%
  group_by(ID) %>% # per subject
  summarise(flanker_score_d1 = mean(flanker_score_list_d1),
            accuracy = mean(accuracy))

LDDM_first_accuracy',s,'<- LDDM_first',s,'_1 %>%
  full_join(LDDM_first',s,'_2, by = "subj_idx") %>%
  full_join(LDDM_first',s,'_3, by="subj_idx") %>%
  group_by(subj_idx, stim) %>% # per subject and condition
  mutate(accuracy_by_stim = sum(response==1)/length(response)) %>%
  summarise(accuracy_by_stim=mean(accuracy_by_stim)) %>%
  pivot_wider(names_from=stim, values_from=accuracy_by_stim, id_cols=subj_idx, names_prefix="accuracy_"
LDDM_second_accuracy',s,'<- LDDM_second',s,'_1 %>%
  full_join(LDDM_second',s,'_2, by = "subj_idx") %>%
  full_join(LDDM_second',s,'_3, by="subj_idx") %>%
  group_by(subj_idx, stim) %>% # per subject and condition
  mutate(accuracy_by_stim = sum(response==1)/length(response)) %>%
  summarise(accuracy_by_stim=mean(accuracy_by_stim)) %>%
  pivot_wider(names_from=stim, values_from=accuracy_by_stim, id_cols=subj_idx, names_prefix="accuracy_"

r_half_',s,' <- cor(LDDM_first',s,'$flanker_score_d1, LDDM_second',s,'$flanker_score_d1, method="spearma
r_half_ci_',s,' <- ci_cor(LDDM_first',s,'$flanker_score_d1, LDDM_second',s,'$flanker_score_d1, method="
r_sb_cilower_',s,' <- (2*r_half_ci_',s,'$interval[1])/(1+r_half_ci_',s,'$interval[1])
```

```
r_sb_ciupper_',s,' <- (2*r_half_ci_',s,'$interval[2])/(1+r_half_ci_',s,'$interval[2])
r_sb_',s,' <- (2*r_half_',s,')/(1+r_half_',s,')

##kr / (1 + (k-1)r)##
# k: Factor by which the length of the test is changed. For example, if original test is 10 questions ar

racc_half_',s,' <- cor(LDDM_first_accuracy',s,'$accuracy_incongruent, LDDM_second_accuracy',s,'$accuracy
racc_half_ci_',s,' <- ci_cor(LDDM_first_accuracy',s,'$accuracy_incongruent, LDDM_second_accuracy',s,'$ac
racc_sb_cilower_',s,' <- (2*racc_half_ci_',s,'$interval[1])/(1+racc_half_ci_',s,'$interval[1])
racc_sb_ciupper_',s,' <- (2*racc_half_ci_',s,'$interval[2])/(1+racc_half_ci_',s,'$interval[2])
racc_sb_',s,' <- (2*racc_half_',s,')/(1+racc_half_',s,')

')))
}
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

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

## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

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

## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

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

```
## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
```

```
## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
```

```
## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
```

```
## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

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

## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

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

## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

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

```
spearman_brown_d1 <- data.frame(trials=seq(15,165,15),
                                rnih=rep(NA,length(seq(15,165,15))),
                                rnih_cilower=rep(NA,length(seq(15,165,15))),
                                rnih_ciupper=rep(NA,length(seq(15,165,15))),
                                racc=rep(NA,length(seq(15,165,15))),
                                racc_cilower=rep(NA,length(seq(15,165,15))),
                                racc_ciupper=rep(NA,length(seq(15,165,15))))
i=1
for (s in seq(15,165,15)){
  eval(parse(text=paste0('spearman_brown_d1[i,2] <- round(as.numeric(r_sb_',s,'),3)
                         spearman_brown_d1[i,3] <- round(as.numeric(r_sb_cilower_',s,'),3)
```

```
                        spearman_brown_d1[i,4] <- round(as.numeric(r_sb_ciupper_',s,'),3)
                        spearman_brown_d1[i,5] <- round(as.numeric(racc_sb_',s,'),3)
                        spearman_brown_d1[i,6] <- round(as.numeric(racc_sb_cilower_',s,'),3)
                        spearman_brown_d1[i,7] <- round(as.numeric(racc_sb_ciupper_',s,'),3)')))
  i=i+1
}

sttr_d1_ddm_splithalf <- read.csv(here("Split_Half/StTr_S1_splithalf.csv"))
spearman_brown_d1$rdriftcon <- sttr_d1_ddm_splithalf$avtz_DO_v_vcon
spearman_brown_d1$rdriftincon <- sttr_d1_ddm_splithalf$avtz_DO_v_vinc
spearman_brown_d1$rboundary_separation <- sttr_d1_ddm_splithalf$avtz_DO_v_a

# ggplot(spearman_brown_d1, aes(x=trials, y=rnih, group=1)) +
#   geom_line() +
#   geom_point() +
#   scale_y_continuous(limits = c(-0.2, 1)) +
#   geom_ribbon(aes(ymin = r_cilower, ymax = r_ciupper), alpha = 0.2)
#
# ggplot(spearman_brown_d1, aes(x=trials, y=racc, group=1)) +
#   geom_line() +
#   geom_point() +
#   scale_y_continuous(limits = c(-0.2, 1)) +
#   geom_ribbon(aes(ymin = racc_cilower, ymax = racc_ciupper), alpha = 0.2)
```

Day 2

```
load(file=here("work/data/LDDM_cleaning04_fullbeh_d2.RData"))

LDDM_cleaning04_d2_reliability <- LDDM_cleaning04_fullbeh_d2 %>%
  group_by(subj_idx) %>%
  mutate(block = c(rep(1,30),rep(2,30),rep(3,30),rep(4,30),rep(5,30),rep(6,30),rep(7,30),rep(8,30),rep(
  group_by(subj_idx) %>%
  mutate(trial = 1:330)

for (s in seq(15,166,15)){
    eval(parse(text=paste0('
LDDM_first',s,'_1 <- LDDM_cleaning04_d2_reliability %>% filter(trial<=',s,')
LDDM_second',s,'_1 <- LDDM_cleaning04_d2_reliability %>% filter(trial< ',(s*2)+1,' & trial>',s,')

LDDM_first',s,'_2 <- LDDM_first',s,'_1 %>%
  group_by(subj_idx) %>% # per subject
  summarise(accuracy_score = sum(response==1)*(5/length(trial))) # accuracy score per NIH TOoolbox manua
LDDM_second',s,'_2 <- LDDM_second',s,'_1 %>%
  group_by(subj_idx) %>%
  summarise(accuracy_score = sum(response==1)*(5/length(trial))) # accuracy score per NIH TOoolbox manua

LDDM_first',s,'_3 <-  LDDM_first',s,'_1 %>%
  group_by(subj_idx) %>% # per subject
  filter(stim=="incongruent" & response==1) %>% # incongruent trials with correct response
  mutate(mean_rt = mean(rt),
         sd_rt = sd(rt)) %>% # compute individual mean and sd RT for use below
```

```
  filter(rt>=0.1 & rt>(mean_rt - 3*sd_rt) & rt<(mean_rt + 3*sd_rt)) %>% # remove trials less than 100ms
  summarise(med_rt = median(rt)*1000) %>% # compute individual level median RT
  mutate(rt_score = 5-(5*((log(med_rt)-log(250))/(log(1000)-log(250))))) # compute RT score to go into
LDDM_second',s,'_3 <-  LDDM_second',s,'_1 %>%
  group_by(subj_idx) %>% # per subject
  filter(stim=="incongruent" & response==1) %>% # incongruent trials with correct response
  mutate(mean_rt = mean(rt),
         sd_rt = sd(rt)) %>% # compute individual mean and sd RT for use below
  filter(rt>=0.1 & rt>(mean_rt - 3*sd_rt) & rt<(mean_rt + 3*sd_rt)) %>% # remove trials less than 100ms
  summarise(med_rt = median(rt)*1000) %>% # compute individual level median RT
  mutate(rt_score = 5-(5*((log(med_rt)-log(250))/(log(1000)-log(250))))) # compute RT score to go into

LDDM_first',s,' <- LDDM_first',s,'_1 %>%
  full_join(LDDM_first',s,'_2, by = "subj_idx") %>%
  full_join(LDDM_first',s,'_3, by="subj_idx") %>%
  group_by(subj_idx) %>% # per subject
  mutate(total_accuracy_perc = sum(response==1)/length(response)) %>% # compute total accuracy percentag
  summarise(flanker_score_list_d2 = if_else(total_accuracy_perc>=0.8, accuracy_score+rt_score, accuracy
            accuracy = mean(total_accuracy_perc)) %>% # if accuracy is above 80% then add accuracy and
  transmute(ID=subj_idx, flanker_score_list_d2=flanker_score_list_d2, accuracy=accuracy) %>%
  group_by(ID) %>% # per subject
  summarise(flanker_score_d2 = mean(flanker_score_list_d2),
            accuracy = mean(accuracy))
LDDM_second',s,' <- LDDM_second',s,'_1 %>%
  full_join(LDDM_second',s,'_2, by = "subj_idx") %>%
  full_join(LDDM_second',s,'_3, by="subj_idx") %>%
  group_by(subj_idx) %>% # per subject
  mutate(total_accuracy_perc = sum(response==1)/length(response)) %>% # compute total accuracy percentag
  summarise(flanker_score_list_d2 = if_else(total_accuracy_perc>=0.8, accuracy_score+rt_score, accuracy
            accuracy = mean(total_accuracy_perc)) %>% # if accuracy is above 80% then add accuracy and
  transmute(ID=subj_idx, flanker_score_list_d2=flanker_score_list_d2, accuracy=accuracy) %>%
  group_by(ID) %>% # per subject
  summarise(flanker_score_d2 = mean(flanker_score_list_d2),
            accuracy = mean(accuracy))

LDDM_first_accuracy',s,'<- LDDM_first',s,'_1 %>%
  full_join(LDDM_first',s,'_2, by = "subj_idx") %>%
  full_join(LDDM_first',s,'_3, by="subj_idx") %>%
  group_by(subj_idx, stim) %>% # per subject and condition
  mutate(accuracy_by_stim = sum(response==1)/length(response)) %>%
  summarise(accuracy_by_stim=mean(accuracy_by_stim)) %>%
  pivot_wider(names_from=stim, values_from=accuracy_by_stim, id_cols=subj_idx, names_prefix="accuracy_"
LDDM_second_accuracy',s,'<- LDDM_second',s,'_1 %>%
  full_join(LDDM_second',s,'_2, by = "subj_idx") %>%
  full_join(LDDM_second',s,'_3, by="subj_idx") %>%
  group_by(subj_idx, stim) %>% # per subject and condition
  mutate(accuracy_by_stim = sum(response==1)/length(response)) %>%
  summarise(accuracy_by_stim=mean(accuracy_by_stim)) %>%
  pivot_wider(names_from=stim, values_from=accuracy_by_stim, id_cols=subj_idx, names_prefix="accuracy_"

r_half_',s,' <- cor(LDDM_first',s,'$flanker_score_d2, LDDM_second',s,'$flanker_score_d2, method="spearma
r_half_ci_',s,' <- ci_cor(LDDM_first',s,'$flanker_score_d2, LDDM_second',s,'$flanker_score_d2, method="
r_sb_cilower_',s,' <- (2*r_half_ci_',s,'$interval[1])/(1+r_half_ci_',s,'$interval[1])
```

```
r_sb_ciupper_',s,' <- (2*r_half_ci_',s,'$interval[2])/(1+r_half_ci_',s,'$interval[2])
r_sb_',s,' <- (2*r_half_',s,')/(1+r_half_',s,')

racc_half_',s,' <- cor(LDDM_first_accuracy',s,'$accuracy_incongruent, LDDM_second_accuracy',s,'$accuracy
racc_half_ci_',s,' <- ci_cor(LDDM_first_accuracy',s,'$accuracy_incongruent, LDDM_second_accuracy',s,'$a
racc_sb_cilower_',s,' <- (2*racc_half_ci_',s,'$interval[1])/(1+racc_half_ci_',s,'$interval[1])
racc_sb_ciupper_',s,' <- (2*racc_half_ci_',s,'$interval[2])/(1+racc_half_ci_',s,'$interval[2])
racc_sb_',s,' <- (2*racc_half_',s,')/(1+racc_half_',s,')

')))
}
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

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

## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

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

## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

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

## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
```

```
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
```

```
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
```

```
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
```

```
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
```

```
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
```

```r
spearman_brown_d2 <- data.frame(trials=seq(15,165,15),
                                rnih=rep(NA,length(seq(15,165,15))),
                                rnih_cilower=rep(NA,length(seq(15,165,15))),
                                rnih_ciupper=rep(NA,length(seq(15,165,15))),
                                racc=rep(NA,length(seq(15,165,15))),
                                racc_cilower=rep(NA,length(seq(15,165,15))),
                                racc_ciupper=rep(NA,length(seq(15,165,15))))
i=1
for (s in seq(15,165,15)){
  eval(parse(text=paste0('spearman_brown_d2[i,2] <- round(as.numeric(r_sb_',s,'),3)
                         spearman_brown_d2[i,3] <- round(as.numeric(r_sb_cilower_',s,'),3)
                         spearman_brown_d2[i,4] <- round(as.numeric(r_sb_ciupper_',s,'),3)
                         spearman_brown_d2[i,5] <- round(as.numeric(racc_sb_',s,'),3)
```

```
                        spearman_brown_d2[i,6] <- round(as.numeric(racc_sb_cilower_',s,'),3)
                        spearman_brown_d2[i,7] <- round(as.numeric(racc_sb_ciupper_',s,'),3)')))
  i=i+1
}

sttr_d2_ddm_splithalf <- read.csv(here("Split_Half/StTr_S2_splithalf.csv"))
spearman_brown_d2$rdriftcon <- sttr_d2_ddm_splithalf$avtz_DO_v_vcon
spearman_brown_d2$rdriftincon <- sttr_d2_ddm_splithalf$avtz_DO_v_vincon
spearman_brown_d2$rboundary_separation <- sttr_d2_ddm_splithalf$avtz_DO_v_a

# spearman_brown_d2
#
# ggplot(spearman_brown_d2, aes(x=trials, y=rnih, group=1)) +
#   geom_line() +
#   geom_point() +
#   scale_y_continuous(limits = c(0, 1)) +
#   geom_ribbon(aes(ymin = r_cilower, ymax = r_ciupper), alpha = 0.2)
```

Day 3

```
# Calculate typical metrics
load(file=here("work/data/LDDM_cleaning04_fullbeh_d3.RData"))

LDDM_cleaning04_d3_calc1 <- LDDM_cleaning04_fullbeh_d3 %>%
  group_by(subj_idx) %>%
  mutate(block = c(rep(1,30),rep(2,30),rep(3,30),rep(4,30),rep(5,30),rep(6,30),rep(7,30),rep(8,30),rep(
  group_by(subj_idx, block) %>%
  mutate(trial = 1:30)

LDDM_cleaning04_d3_reliability <- LDDM_cleaning04_fullbeh_d3 %>%
  group_by(subj_idx) %>%
  mutate(block = c(rep(1,30),rep(2,30),rep(3,30),rep(4,30),rep(5,30),rep(6,30),rep(7,30),rep(8,30),rep(
  group_by(subj_idx) %>%
  mutate(trial = 1:330)

for (s in seq(15,166,15)){
    eval(parse(text=paste0('
LDDM_first',s,'_1 <- LDDM_cleaning04_d3_reliability %>% filter(trial<=',s,')
LDDM_second',s,'_1 <- LDDM_cleaning04_d3_reliability %>% filter(trial< ',(s*2)+1,' & trial>',s,')

LDDM_first',s,'_2 <- LDDM_first',s,'_1 %>%
  group_by(subj_idx) %>% # per subject
  summarise(accuracy_score = sum(response==1)*(5/length(trial))) # accuracy score per NIH TOoolbox manua
LDDM_second',s,'_2 <- LDDM_second',s,'_1 %>%
  group_by(subj_idx) %>%
  summarise(accuracy_score = sum(response==1)*(5/length(trial))) # accuracy score per NIH TOoolbox manua

LDDM_first',s,'_3 <-  LDDM_first',s,'_1 %>%
  group_by(subj_idx) %>% # per subject
  filter(stim=="incongruent" & response==1) %>% # incongruent trials with correct response
  mutate(mean_rt = mean(rt),
         sd_rt = sd(rt)) %>% # compute individual mean and sd RT for use below
```

```
  filter(rt>=0.1 & rt>(mean_rt - 3*sd_rt) & rt<(mean_rt + 3*sd_rt)) %>% # remove trials less than 100ms
  summarise(med_rt = median(rt)*1000) %>% # compute individual level median RT
  mutate(rt_score = 5-(5*((log(med_rt)-log(250))/(log(1000)-log(250))))) # compute RT score to go into
LDDM_second',s,'_3 <-  LDDM_second',s,'_1 %>%
  group_by(subj_idx) %>% # per subject
  filter(stim=="incongruent" & response==1) %>% # incongruent trials with correct response
  mutate(mean_rt = mean(rt),
         sd_rt = sd(rt)) %>% # compute individual mean and sd RT for use below
  filter(rt>=0.1 & rt>(mean_rt - 3*sd_rt) & rt<(mean_rt + 3*sd_rt)) %>% # remove trials less than 100ms
  summarise(med_rt = median(rt)*1000) %>% # compute individual level median RT
  mutate(rt_score = 5-(5*((log(med_rt)-log(250))/(log(1000)-log(250))))) # compute RT score to go into

LDDM_first',s,' <- LDDM_first',s,'_1 %>%
  full_join(LDDM_first',s,'_2, by = "subj_idx") %>%
  full_join(LDDM_first',s,'_3, by="subj_idx") %>%
  group_by(subj_idx) %>% # per subject
  mutate(total_accuracy_perc = sum(response==1)/length(response)) %>% # compute total accuracy percentag
  summarise(flanker_score_list_d3 = if_else(total_accuracy_perc>=0.8, accuracy_score+rt_score, accuracy
            accuracy = mean(total_accuracy_perc)) %>% # if accuracy is above 80% then add accuracy and
  transmute(ID=subj_idx, flanker_score_list_d3=flanker_score_list_d3, accuracy=accuracy) %>%
  group_by(ID) %>% # per subject
  summarise(flanker_score_d3 = mean(flanker_score_list_d3),
            accuracy = mean(accuracy))
LDDM_second',s,' <- LDDM_second',s,'_1 %>%
  full_join(LDDM_second',s,'_2, by = "subj_idx") %>%
  full_join(LDDM_second',s,'_3, by="subj_idx") %>%
  group_by(subj_idx) %>% # per subject
  mutate(total_accuracy_perc = sum(response==1)/length(response)) %>% # compute total accuracy percentag
  summarise(flanker_score_list_d3 = if_else(total_accuracy_perc>=0.8, accuracy_score+rt_score, accuracy
            accuracy = mean(total_accuracy_perc)) %>% # if accuracy is above 80% then add accuracy and
  transmute(ID=subj_idx, flanker_score_list_d3=flanker_score_list_d3, accuracy=accuracy) %>%
  group_by(ID) %>% # per subject
  summarise(flanker_score_d3 = mean(flanker_score_list_d3),
            accuracy = mean(accuracy))

LDDM_first_accuracy',s,'<- LDDM_first',s,'_1 %>%
  full_join(LDDM_first',s,'_2, by = "subj_idx") %>%
  full_join(LDDM_first',s,'_3, by="subj_idx") %>%
  group_by(subj_idx, stim) %>% # per subject and condition
  mutate(accuracy_by_stim = sum(response==1)/length(response)) %>%
  summarise(accuracy_by_stim=mean(accuracy_by_stim)) %>%
  pivot_wider(names_from=stim, values_from=accuracy_by_stim, id_cols=subj_idx, names_prefix="accuracy_"
LDDM_second_accuracy',s,'<- LDDM_second',s,'_1 %>%
  full_join(LDDM_second',s,'_2, by = "subj_idx") %>%
  full_join(LDDM_second',s,'_3, by="subj_idx") %>%
  group_by(subj_idx, stim) %>% # per subject and condition
  mutate(accuracy_by_stim = sum(response==1)/length(response)) %>%
  summarise(accuracy_by_stim=mean(accuracy_by_stim)) %>%
  pivot_wider(names_from=stim, values_from=accuracy_by_stim, id_cols=subj_idx, names_prefix="accuracy_"

r_half_',s,' <- cor(LDDM_first',s,'$flanker_score_d3, LDDM_second',s,'$flanker_score_d3, method="spearma
r_half_ci_',s,' <- ci_cor(LDDM_first',s,'$flanker_score_d3, LDDM_second',s,'$flanker_score_d3, method="
r_sb_cilower_',s,' <- (2*r_half_ci_',s,'$interval[1])/(1+r_half_ci_',s,'$interval[1])
```

```
r_sb_ciupper_',s,' <- (2*r_half_ci_',s,'$interval[2])/(1+r_half_ci_',s,'$interval[2])
r_sb_',s,' <- (2*r_half_',s,')/(1+r_half_',s,')

racc_half_',s,' <- cor(LDDM_first_accuracy',s,'$accuracy_incongruent, LDDM_second_accuracy',s,'$accuracy
racc_half_ci_',s,' <- ci_cor(LDDM_first_accuracy',s,'$accuracy_incongruent, LDDM_second_accuracy',s,'$a
racc_sb_cilower_',s,' <- (2*racc_half_ci_',s,'$interval[1])/(1+racc_half_ci_',s,'$interval[1])
racc_sb_ciupper_',s,' <- (2*racc_half_ci_',s,'$interval[2])/(1+racc_half_ci_',s,'$interval[2])
racc_sb_',s,' <- (2*racc_half_',s,')/(1+racc_half_',s,')

')))
}
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

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

## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

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

## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

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

## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
```

```
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
```

```
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
```

```
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
```

```
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
```

```
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.

## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
## ‘summarise()‘ has grouped output by ’subj_idx’. You can override using the
## ‘.groups‘ argument.
```

```
spearman_brown_d3 <- data.frame(trials=seq(15,165,15),
                                rnih=rep(NA,length(seq(15,165,15))),
                                rnih_cilower=rep(NA,length(seq(15,165,15))),
                                rnih_ciupper=rep(NA,length(seq(15,165,15))),
                                racc=rep(NA,length(seq(15,165,15))),
                                racc_cilower=rep(NA,length(seq(15,165,15))),
                                racc_ciupper=rep(NA,length(seq(15,165,15))))
i=1
for (s in seq(15,165,15)){
  eval(parse(text=paste0('spearman_brown_d3[i,2] <- round(as.numeric(r_sb_',s,'),3)
                         spearman_brown_d3[i,3] <- round(as.numeric(r_sb_cilower_',s,'),3)
                         spearman_brown_d3[i,4] <- round(as.numeric(r_sb_ciupper_',s,'),3)
                         spearman_brown_d3[i,5] <- round(as.numeric(racc_sb_',s,'),3)
```

```
                        spearman_brown_d3[i,6] <- round(as.numeric(racc_sb_cilower_',s,'),3)
                        spearman_brown_d3[i,7] <- round(as.numeric(racc_sb_ciupper_',s,'),3)')))
  i=i+1
}
sttr_d3_ddm_splithalf <- read.csv(here("Split_Half/StTr_S3_splithalf.csv"))
spearman_brown_d3$rdriftcon <- sttr_d3_ddm_splithalf$avtz_DO_v_vcon
spearman_brown_d3$rdriftincon <- sttr_d3_ddm_splithalf$avtz_DO_v_vincon
spearman_brown_d3$rboundary_separation <- sttr_d3_ddm_splithalf$avtz_DO_v_a

# ggplot(spearman_brown_d3, aes(x=trials, y=rnih, group=1)) +
#   geom_line() +
#   geom_point() +
#   scale_y_continuous(limits = c(0, 1)) +
#   geom_ribbon(aes(ymin = r_cilower, ymax = r_ciupper), alpha = 0.2)
```

Saving datasets

In this step, go ahead and close out of the file and quit R without saving the work space.

```
save(LDDM_do2_d1_not_outliers, file=here("work/data/LDDM_do2_d1_not_outliers.RData"))
save(LDDM_do2_d2_not_outliers, file=here("work/data/LDDM_do2_d2_not_outliers.RData"))
save(LDDM_do2_d3_not_outliers, file=here("work/data/LDDM_do2_d3_not_outliers.RData"))

# save(LDDM_cleaning04_calc3_1to3, file=here("work/data/LDDM_cleaning04_calc3_1to3.RData"))
save(LDDM_do2_irr, file=here("work/data/LDDM_do2_irr.RData"))
# save(LDDM_do2, file=here("work/data/LDDM_do2.RData"))
save(spearman_brown_d1, file=here("work/data/spearman_brown_d1.RData"))
save(spearman_brown_d2, file=here("work/data/spearman_brown_d2.RData"))
save(spearman_brown_d3, file=here("work/data/spearman_brown_d3.RData"))
```