Puplishable script for ATTENTION paper

Rasmus Ahmt & Marta Topor

2023-3-22

## function that change "," to ". https://stackoverflow.com/questions/59380129/changing-to-within-a-function-in-r  
convert\_number <- function(x) {  
 x <- as.character(x)  
 x <- gsub(  
 pattern = ",",  
 replacement = ".",  
 x = x,  
 fixed = TRUE  
 )  
 x <- as.numeric(x)  
 return(x)  
}  
##  
  
# This section of the script is responsible for importing data from the designated folder.  
# The purpose of importing it into a long and tidy format is to ensure that the data can be easily manipulated and analyzed using tidyverse.  
  
#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_VR\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#  
# Firstly i wish to import all the VR data.  
# sett working directory  
setwd(  
 "/Users/rasmus/Desktop/Playore VR data/mellemskridt/Attention\_data/VR attention data RAW"  
)  
# Instruct R to look in the directory and search for CSV files.  
list <- list.files(pattern = ".csv", recursive = TRUE)  
# "i" represents the current participant and is set to loop through all files in the directory.  
# It is set to loop from 1 to the length of the list of files in our directory.  
i <- 1:length(list)  
  
  
  
DF <- data.frame(  
 id = character(),  
 incongruent\_and\_audio\_RT = numeric(),  
 Correct\_answers\_incongruent\_and\_audio = numeric(),  
 incongruent\_and\_visual\_RT = numeric(),  
 Correct\_answers\_incongruent\_and\_visual = numeric(),  
 incongruent\_and\_audio.visual\_RT = numeric(),  
 Correct\_answers\_incongruent\_and\_audio.visual = numeric(),  
 congruent\_and\_audio\_RT = numeric(),  
 Correct\_answers\_congruent\_and\_audio = numeric(),  
 congruent\_and\_visual\_RT = numeric(),  
 Correct\_answers\_congruent\_and\_visual = numeric(),  
 congruent\_and\_audio.visual\_RT = numeric(),  
 Correct\_answers\_congruent\_and\_audio.visual = numeric(),  
 neutral\_and\_audio\_RT = numeric(),  
 Correct\_answers\_neutral\_and\_audio = numeric(),  
 neutral\_and\_visual\_RT = numeric(),  
 Correct\_answers\_neutral\_and\_visual = numeric(),  
 neutral\_and\_audio.visual\_RT = numeric(),  
 Correct\_answers\_neutral\_and\_audio.visual = numeric(),  
 stringsAsFactors = FALSE  
)  
  
  
  
# settup import and extract loop  
for (i in list)  
{  
 myData <- read.csv2(i)  
   
 ## inconguent  
 # incongruent\_and\_audio  
 incongruent\_and\_audio1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Incongruent" &  
 Distractor.Modality.Block1 == "audio"  
 )  
 incongruent\_and\_audio <-  
 convert\_number(incongruent\_and\_audio1$ReactionTime.Block1)  
   
 if (is\_empty(incongruent\_and\_audio) == TRUE) {  
 incongruent\_and\_audio\_RT = 0  
 } else{  
 incongruent\_and\_audio\_RT <- mean(incongruent\_and\_audio)  
 }  
 if (is\_empty(incongruent\_and\_audio) == TRUE) {  
 Correct\_answers\_incongruent\_and\_audio = 0  
 } else{  
 Correct\_answers\_incongruent\_and\_audio <-  
 length(incongruent\_and\_audio1$Answers.Block1)  
 }  
   
 # incongruent\_and\_visual  
 incongruent\_and\_visual1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Incongruent" &  
 Distractor.Modality.Block1 == "visual"  
 )  
 incongruent\_and\_visual <-  
 convert\_number(incongruent\_and\_visual1$ReactionTime.Block1)  
   
 if (is\_empty(incongruent\_and\_visual) == TRUE) {  
 incongruent\_and\_visual\_RT = 0  
 } else{  
 incongruent\_and\_visual\_RT <- mean(incongruent\_and\_visual)  
 }  
 if (is\_empty(incongruent\_and\_visual) == TRUE) {  
 Correct\_answers\_incongruent\_and\_visual = 0  
 } else{  
 Correct\_answers\_incongruent\_and\_visual <-  
 length(incongruent\_and\_visual1$Answers.Block1)  
 }  
   
 # incongruent\_and\_visual and audio  
 incongruent\_and\_audio.visual1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Incongruent" &  
 Distractor.Modality.Block1 == "audio visual"  
 )  
 incongruent\_and\_audio.visual <-  
 convert\_number(incongruent\_and\_audio.visual1$ReactionTime.Block1)  
   
 if (is\_empty(incongruent\_and\_audio.visual) == TRUE) {  
 incongruent\_and\_audio.visual\_RT = 0  
 } else{  
 incongruent\_and\_audio.visual\_RT <-  
 mean(incongruent\_and\_audio.visual)  
 }  
 if (is\_empty(incongruent\_and\_audio.visual) == TRUE) {  
 Correct\_answers\_incongruent\_and\_audio.visual = 0  
 } else{  
 Correct\_answers\_incongruent\_and\_audio.visual <-  
 length(incongruent\_and\_audio.visual1$Answers.Block1)  
 }  
   
 ## congruent  
 # congruent\_and\_audio  
 congruent\_and\_audio1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Congruent" &  
 Distractor.Modality.Block1 == "audio"  
 )  
 congruent\_and\_audio <-  
 convert\_number(congruent\_and\_audio1$ReactionTime.Block1)  
   
 if (is\_empty(congruent\_and\_audio) == TRUE) {  
 congruent\_and\_audio\_RT = 0  
 } else{  
 congruent\_and\_audio\_RT <- mean(congruent\_and\_audio)  
 }  
 if (is\_empty(congruent\_and\_audio) == TRUE) {  
 Correct\_answers\_congruent\_and\_audio = 0  
 } else{  
 Correct\_answers\_congruent\_and\_audio <-  
 length(congruent\_and\_audio1$Answers.Block1)  
 }  
   
 # congruent\_and\_visual  
 congruent\_and\_visual1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Congruent" &  
 Distractor.Modality.Block1 == "visual"  
 )  
 congruent\_and\_visual <-  
 convert\_number(congruent\_and\_visual1$ReactionTime.Block1)  
   
 if (is\_empty(congruent\_and\_visual) == TRUE) {  
 congruent\_and\_visual\_RT = 0  
 } else{  
 congruent\_and\_visual\_RT <- mean(congruent\_and\_visual)  
 }  
 if (is\_empty(congruent\_and\_visual) == TRUE) {  
 Correct\_answers\_congruent\_and\_visual = 0  
 } else{  
 Correct\_answers\_congruent\_and\_visual <-  
 length(congruent\_and\_visual1$Answers.Block1)  
 }  
   
 # congruent\_and\_visual and audio  
 congruent\_and\_audio.visual1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Congruent" &  
 Distractor.Modality.Block1 == "audio visual"  
 )  
 congruent\_and\_audio.visual <-  
 convert\_number(congruent\_and\_audio.visual1$ReactionTime.Block1)  
   
 if (is\_empty(congruent\_and\_audio.visual) == TRUE) {  
 congruent\_and\_audio.visual\_RT = 0  
 } else{  
 congruent\_and\_audio.visual\_RT <- mean(congruent\_and\_audio.visual)  
 }  
 if (is\_empty(congruent\_and\_audio.visual) == TRUE) {  
 Correct\_answers\_congruent\_and\_audio.visual = 0  
 } else{  
 Correct\_answers\_congruent\_and\_audio.visual <-  
 length(congruent\_and\_audio.visual1$Answers.Block1)  
 }  
   
 ## neutral  
 # neutral\_and\_audio  
 neutral\_and\_audio1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Neutral" &  
 Distractor.Modality.Block1 == "audio"  
 )  
 neutral\_and\_audio <-  
 convert\_number(neutral\_and\_audio1$ReactionTime.Block1)  
   
 if (is\_empty(neutral\_and\_audio) == TRUE) {  
 neutral\_and\_audio\_RT = 0  
 } else{  
 neutral\_and\_audio\_RT <- mean(neutral\_and\_audio)  
 }  
 if (is\_empty(neutral\_and\_audio) == TRUE) {  
 Correct\_answers\_neutral\_and\_audio = 0  
 } else{  
 Correct\_answers\_neutral\_and\_audio <-  
 length(neutral\_and\_audio1$Answers.Block1)  
 }  
   
 # neutral\_and\_visual  
 neutral\_and\_visual1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Neutral" &  
 Distractor.Modality.Block1 == "visual"  
 )  
 neutral\_and\_visual <-  
 convert\_number(neutral\_and\_visual1$ReactionTime.Block1)  
   
 if (is\_empty(neutral\_and\_visual) == TRUE) {  
 neutral\_and\_visual\_RT = 0  
 } else{  
 neutral\_and\_visual\_RT <- mean(neutral\_and\_visual)  
 }  
 if (is\_empty(neutral\_and\_visual) == TRUE) {  
 Correct\_answers\_neutral\_and\_visual = 0  
 } else{  
 Correct\_answers\_neutral\_and\_visual <-  
 length(neutral\_and\_visual1$Answers.Block1)  
 }  
   
 # neutral\_and\_visual and audio  
 neutral\_and\_audio.visual1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Neutral" &  
 Distractor.Modality.Block1 == "audio visual"  
 )  
 neutral\_and\_audio.visual <-  
 convert\_number(neutral\_and\_audio.visual1$ReactionTime.Block1)  
   
 if (is\_empty(neutral\_and\_audio.visual) == TRUE) {  
 neutral\_and\_audio.visual\_RT = 0  
 } else{  
 neutral\_and\_audio.visual\_RT <- mean(neutral\_and\_audio.visual)  
 }  
 if (is\_empty(neutral\_and\_audio.visual) == TRUE) {  
 Correct\_answers\_neutral\_and\_audio.visual = 0  
 } else{  
 Correct\_answers\_neutral\_and\_audio.visual <-  
 length(neutral\_and\_audio.visual1$Answers.Block1)  
 }  
   
 DF[i, ] <-  
 data.frame(  
 i,  
 incongruent\_and\_audio\_RT,  
 Correct\_answers\_incongruent\_and\_audio,  
 incongruent\_and\_visual\_RT,  
 Correct\_answers\_incongruent\_and\_visual,  
 incongruent\_and\_audio.visual\_RT,  
 Correct\_answers\_incongruent\_and\_audio.visual,  
 congruent\_and\_audio\_RT,  
 Correct\_answers\_congruent\_and\_audio,  
 congruent\_and\_visual\_RT,  
 Correct\_answers\_congruent\_and\_visual,  
 congruent\_and\_audio.visual\_RT,  
 Correct\_answers\_congruent\_and\_audio.visual,  
 neutral\_and\_audio\_RT,  
 Correct\_answers\_neutral\_and\_audio,  
 neutral\_and\_visual\_RT,  
 Correct\_answers\_neutral\_and\_visual,  
 neutral\_and\_audio.visual\_RT,  
 Correct\_answers\_neutral\_and\_audio.visual  
 )  
   
}  
# specefy a new data sett thats a temporary data set to hold the information from the VR data, so we can reuse this loop for the PC files.  
VR\_not\_done <- DF  
  
  
#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_PC\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#  
# Secondly i wish to import all the PC data.  
# sett working directory  
setwd(  
 "/Users/rasmus/Desktop/Playore VR data/mellemskridt/Attention\_data/PC attention data RAW"  
)  
# Instruct R to look in the directory and search for CSV files.  
list <- list.files(pattern = ".csv", recursive = TRUE)  
# "i" represents the current participant and is set to loop through all files in the directory.  
# It is set to loop from 1 to the length of the list of files in our directory.  
i <- 1:length(list)  
  
  
  
DF <- data.frame(  
 id = character(),  
 incongruent\_and\_audio\_RT = numeric(),  
 Correct\_answers\_incongruent\_and\_audio = numeric(),  
 incongruent\_and\_visual\_RT = numeric(),  
 Correct\_answers\_incongruent\_and\_visual = numeric(),  
 incongruent\_and\_audio.visual\_RT = numeric(),  
 Correct\_answers\_incongruent\_and\_audio.visual = numeric(),  
 congruent\_and\_audio\_RT = numeric(),  
 Correct\_answers\_congruent\_and\_audio = numeric(),  
 congruent\_and\_visual\_RT = numeric(),  
 Correct\_answers\_congruent\_and\_visual = numeric(),  
 congruent\_and\_audio.visual\_RT = numeric(),  
 Correct\_answers\_congruent\_and\_audio.visual = numeric(),  
 neutral\_and\_audio\_RT = numeric(),  
 Correct\_answers\_neutral\_and\_audio = numeric(),  
 neutral\_and\_visual\_RT = numeric(),  
 Correct\_answers\_neutral\_and\_visual = numeric(),  
 neutral\_and\_audio.visual\_RT = numeric(),  
 Correct\_answers\_neutral\_and\_audio.visual = numeric(),  
 stringsAsFactors = FALSE  
)  
  
  
  
  
# settup import and extract loop  
for (i in list)  
{  
 myData <- read.csv2(i)  
   
 ## inconguent  
 # incongruent\_and\_audio  
 incongruent\_and\_audio1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Incongruent" &  
 Distractor.Modality.Block1 == "audio"  
 )  
 incongruent\_and\_audio <-  
 convert\_number(incongruent\_and\_audio1$ReactionTime.Block1)  
   
 if (is\_empty(incongruent\_and\_audio) == TRUE) {  
 incongruent\_and\_audio\_RT = 0  
 } else{  
 incongruent\_and\_audio\_RT <- mean(incongruent\_and\_audio)  
 }  
 if (is\_empty(incongruent\_and\_audio) == TRUE) {  
 Correct\_answers\_incongruent\_and\_audio = 0  
 } else{  
 Correct\_answers\_incongruent\_and\_audio <-  
 length(incongruent\_and\_audio1$Answers.Block1)  
 }  
   
 # incongruent\_and\_visual  
 incongruent\_and\_visual1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Incongruent" &  
 Distractor.Modality.Block1 == "visual"  
 )  
 incongruent\_and\_visual <-  
 convert\_number(incongruent\_and\_visual1$ReactionTime.Block1)  
   
 if (is\_empty(incongruent\_and\_visual) == TRUE) {  
 incongruent\_and\_visual\_RT = 0  
 } else{  
 incongruent\_and\_visual\_RT <- mean(incongruent\_and\_visual)  
 }  
 if (is\_empty(incongruent\_and\_visual) == TRUE) {  
 Correct\_answers\_incongruent\_and\_visual = 0  
 } else{  
 Correct\_answers\_incongruent\_and\_visual <-  
 length(incongruent\_and\_visual1$Answers.Block1)  
 }  
   
 # incongruent\_and\_visual and audio  
 incongruent\_and\_audio.visual1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Incongruent" &  
 Distractor.Modality.Block1 == "audio visual"  
 )  
 incongruent\_and\_audio.visual <-  
 convert\_number(incongruent\_and\_audio.visual1$ReactionTime.Block1)  
   
 if (is\_empty(incongruent\_and\_audio.visual) == TRUE) {  
 incongruent\_and\_audio.visual\_RT = 0  
 } else{  
 incongruent\_and\_audio.visual\_RT <-  
 mean(incongruent\_and\_audio.visual)  
 }  
 if (is\_empty(incongruent\_and\_audio.visual) == TRUE) {  
 Correct\_answers\_incongruent\_and\_audio.visual = 0  
 } else{  
 Correct\_answers\_incongruent\_and\_audio.visual <-  
 length(incongruent\_and\_audio.visual1$Answers.Block1)  
 }  
   
 ## congruent  
 # congruent\_and\_audio  
 congruent\_and\_audio1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Congruent" &  
 Distractor.Modality.Block1 == "audio"  
 )  
 congruent\_and\_audio <-  
 convert\_number(congruent\_and\_audio1$ReactionTime.Block1)  
   
 if (is\_empty(congruent\_and\_audio) == TRUE) {  
 congruent\_and\_audio\_RT = 0  
 } else{  
 congruent\_and\_audio\_RT <- mean(congruent\_and\_audio)  
 }  
 if (is\_empty(congruent\_and\_audio) == TRUE) {  
 Correct\_answers\_congruent\_and\_audio = 0  
 } else{  
 Correct\_answers\_congruent\_and\_audio <-  
 length(congruent\_and\_audio1$Answers.Block1)  
 }  
   
 # congruent\_and\_visual  
 congruent\_and\_visual1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Congruent" &  
 Distractor.Modality.Block1 == "visual"  
 )  
 congruent\_and\_visual <-  
 convert\_number(congruent\_and\_visual1$ReactionTime.Block1)  
   
 if (is\_empty(congruent\_and\_visual) == TRUE) {  
 congruent\_and\_visual\_RT = 0  
 } else{  
 congruent\_and\_visual\_RT <- mean(congruent\_and\_visual)  
 }  
 if (is\_empty(congruent\_and\_visual) == TRUE) {  
 Correct\_answers\_congruent\_and\_visual = 0  
 } else{  
 Correct\_answers\_congruent\_and\_visual <-  
 length(congruent\_and\_visual1$Answers.Block1)  
 }  
   
 # congruent\_and\_visual and audio  
 congruent\_and\_audio.visual1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Congruent" &  
 Distractor.Modality.Block1 == "audio visual"  
 )  
 congruent\_and\_audio.visual <-  
 convert\_number(congruent\_and\_audio.visual1$ReactionTime.Block1)  
   
 if (is\_empty(congruent\_and\_audio.visual) == TRUE) {  
 congruent\_and\_audio.visual\_RT = 0  
 } else{  
 congruent\_and\_audio.visual\_RT <- mean(congruent\_and\_audio.visual)  
 }  
 if (is\_empty(congruent\_and\_audio.visual) == TRUE) {  
 Correct\_answers\_congruent\_and\_audio.visual = 0  
 } else{  
 Correct\_answers\_congruent\_and\_audio.visual <-  
 length(congruent\_and\_audio.visual1$Answers.Block1)  
 }  
   
 ## neutral  
 # neutral\_and\_audio  
 neutral\_and\_audio1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Neutral" &  
 Distractor.Modality.Block1 == "audio"  
 )  
 neutral\_and\_audio <-  
 convert\_number(neutral\_and\_audio1$ReactionTime.Block1)  
   
 if (is\_empty(neutral\_and\_audio) == TRUE) {  
 neutral\_and\_audio\_RT = 0  
 } else{  
 neutral\_and\_audio\_RT <- mean(neutral\_and\_audio)  
 }  
 if (is\_empty(neutral\_and\_audio) == TRUE) {  
 Correct\_answers\_neutral\_and\_audio = 0  
 } else{  
 Correct\_answers\_neutral\_and\_audio <-  
 length(neutral\_and\_audio1$Answers.Block1)  
 }  
   
 # neutral\_and\_visual  
 neutral\_and\_visual1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Neutral" &  
 Distractor.Modality.Block1 == "visual"  
 )  
 neutral\_and\_visual <-  
 convert\_number(neutral\_and\_visual1$ReactionTime.Block1)  
   
 if (is\_empty(neutral\_and\_visual) == TRUE) {  
 neutral\_and\_visual\_RT = 0  
 } else{  
 neutral\_and\_visual\_RT <- mean(neutral\_and\_visual)  
 }  
 if (is\_empty(neutral\_and\_visual) == TRUE) {  
 Correct\_answers\_neutral\_and\_visual = 0  
 } else{  
 Correct\_answers\_neutral\_and\_visual <-  
 length(neutral\_and\_visual1$Answers.Block1)  
 }  
   
 # neutral\_and\_visual and audio  
 neutral\_and\_audio.visual1 <-  
 subset(  
 myData,  
 Answer.Codes.Block1 == "1" &  
 Presented.Condition.Block1 == "Neutral" &  
 Distractor.Modality.Block1 == "audio visual"  
 )  
 neutral\_and\_audio.visual <-  
 convert\_number(neutral\_and\_audio.visual1$ReactionTime.Block1)  
   
 if (is\_empty(neutral\_and\_audio.visual) == TRUE) {  
 neutral\_and\_audio.visual\_RT = 0  
 } else{  
 neutral\_and\_audio.visual\_RT <- mean(neutral\_and\_audio.visual)  
 }  
 if (is\_empty(neutral\_and\_audio.visual) == TRUE) {  
 Correct\_answers\_neutral\_and\_audio.visual = 0  
 } else{  
 Correct\_answers\_neutral\_and\_audio.visual <-  
 length(neutral\_and\_audio.visual1$Answers.Block1)  
 }  
   
 DF[i, ] <-  
 data.frame(  
 i,  
 incongruent\_and\_audio\_RT,  
 Correct\_answers\_incongruent\_and\_audio,  
 incongruent\_and\_visual\_RT,  
 Correct\_answers\_incongruent\_and\_visual,  
 incongruent\_and\_audio.visual\_RT,  
 Correct\_answers\_incongruent\_and\_audio.visual,  
 congruent\_and\_audio\_RT,  
 Correct\_answers\_congruent\_and\_audio,  
 congruent\_and\_visual\_RT,  
 Correct\_answers\_congruent\_and\_visual,  
 congruent\_and\_audio.visual\_RT,  
 Correct\_answers\_congruent\_and\_audio.visual,  
 neutral\_and\_audio\_RT,  
 Correct\_answers\_neutral\_and\_audio,  
 neutral\_and\_visual\_RT,  
 Correct\_answers\_neutral\_and\_visual,  
 neutral\_and\_audio.visual\_RT,  
 Correct\_answers\_neutral\_and\_audio.visual  
 )  
   
}  
# we create another temp file for for PC aswell  
PC\_not\_done <- DF

# Firstly we want to get rid of those file names in the id collumn and use the participants actually id names.   
ID <- c(10103, 10106, 10107, 10108, 10110, 10113, 10114, 10116, 10117, 10123, 10124, 10126, 10127, 10128, 10129, 10131, 10133, 10134, 10136, 10137, 10138, 10139, 10153, 10154, 10156, 10157, 10158, 20002, 20003, 20005, 20006, 20008, 20009, 20020, 20021, 20022, 30002, 30003, 30004, 30006, 30009, 30010, 30012, 30013, 30019, 30020, 30022, 30023, 30024, 30025, 30026, 30027, 30028, 30030, 30032, 30033, 30034, 30035, 30036, 30040, 30042, 30044, 30046, 30048, 30049, 30051, 30052, 30053, 30054, 30055, 30058, 30060, 30061)  
PC\_not\_done <- mutate(PC\_not\_done, id = ID)  
VR\_not\_done <- mutate(VR\_not\_done, id = ID)  
  
#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_PC\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#  
# create a empty data frame that's empty ready to fill it up  
LONG\_PC\_DATA <- data.frame(  
 id = character(),  
 stimuli\_type = character(),  
 stimuli\_compatibility = character(),  
 RT = numeric(),  
 corrects = numeric(),  
 stringsAsFactors = FALSE  
)  
  
# set up a loop thats take every ID and execute the loop.  
unique\_ID <- unique(PC\_not\_done$id)  
for (i in seq\_along(unique\_ID)) {  
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(PC\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$incongruent\_and\_audio\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <- participant\_data$Correct\_answers\_incongruent\_and\_audio  
 # define the stimuli type  
 stimuli\_type <- "audio"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "incongurent"  
 # add the new row to the data frame  
 LONG\_PC\_DATA <-  
 rbind(LONG\_PC\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
   
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(PC\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$incongruent\_and\_visual\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <-  
 participant\_data$Correct\_answers\_incongruent\_and\_visual  
 # define the stimuli type  
 stimuli\_type <- "visual"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "incongurent"  
 # add the new row to the data frame  
 LONG\_PC\_DATA <-  
 rbind(LONG\_PC\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
   
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(PC\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$incongruent\_and\_audio.visual\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <-  
 participant\_data$Correct\_answers\_incongruent\_and\_audio.visual  
 # define the stimuli type  
 stimuli\_type <- "audio.visual"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "incongurent"  
 # add the new row to the data frame  
 LONG\_PC\_DATA <-  
 rbind(LONG\_PC\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
   
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(PC\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$congruent\_and\_audio\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <- participant\_data$Correct\_answers\_congruent\_and\_audio  
 # define the stimuli type  
 stimuli\_type <- "audio"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "congruent"  
 # add the new row to the data frame  
 LONG\_PC\_DATA <-  
 rbind(LONG\_PC\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
   
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(PC\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$congruent\_and\_visual\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <- participant\_data$Correct\_answers\_congruent\_and\_visual  
 # define the stimuli type  
 stimuli\_type <- "visual"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "congruent"  
 # add the new row to the data frame  
 LONG\_PC\_DATA <-  
 rbind(LONG\_PC\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
   
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(PC\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$congruent\_and\_audio.visual\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <-  
 participant\_data$Correct\_answers\_congruent\_and\_audio.visual  
 # define the stimuli type  
 stimuli\_type <- "audio.visual"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "congruent"  
 # add the new row to the data frame  
 LONG\_PC\_DATA <-  
 rbind(LONG\_PC\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
   
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(PC\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$neutral\_and\_audio\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <- participant\_data$Correct\_answers\_neutral\_and\_audio  
 # define the stimuli type  
 stimuli\_type <- "audio"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "neutral"  
 # add the new row to the data frame  
 LONG\_PC\_DATA <-  
 rbind(LONG\_PC\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
   
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(PC\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$neutral\_and\_visual\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <- participant\_data$Correct\_answers\_neutral\_and\_visual  
 # define the stimuli type  
 stimuli\_type <- "visual"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "neutral"  
 # add the new row to the data frame  
 LONG\_PC\_DATA <-  
 rbind(LONG\_PC\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
   
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(PC\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$neutral\_and\_audio.visual\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <-  
 participant\_data$Correct\_answers\_neutral\_and\_audio.visual  
 # define the stimuli type  
 stimuli\_type <- "audio.visual"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "neutral"  
 # add the new row to the data frame  
 LONG\_PC\_DATA <-  
 rbind(LONG\_PC\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
}  
  
#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_VR\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#  
# create a empty data frame that's empty ready to fill it up  
LONG\_VR\_DATA <- data.frame(  
 id = character(),  
 stimuli\_type = character(),  
 stimuli\_compatibility = character(),  
 RT = numeric(),  
 corrects = numeric(),  
 stringsAsFactors = FALSE  
)  
  
# set up a loop thats take every ID and execute the loop.  
unique\_ID <- unique(VR\_not\_done$id)  
for (i in seq\_along(unique\_ID)) {  
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(VR\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$incongruent\_and\_audio\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <- participant\_data$Correct\_answers\_incongruent\_and\_audio  
 # define the stimuli type  
 stimuli\_type <- "audio"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "incongurent"  
 # add the new row to the data frame  
 LONG\_VR\_DATA <-  
 rbind(LONG\_VR\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
   
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(VR\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$incongruent\_and\_visual\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <-  
 participant\_data$Correct\_answers\_incongruent\_and\_visual  
 # define the stimuli type  
 stimuli\_type <- "visual"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "incongurent"  
 # add the new row to the data frame  
 LONG\_VR\_DATA <-  
 rbind(LONG\_VR\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
   
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(VR\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$incongruent\_and\_audio.visual\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <-  
 participant\_data$Correct\_answers\_incongruent\_and\_audio.visual  
 # define the stimuli type  
 stimuli\_type <- "audio.visual"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "incongurent"  
 # add the new row to the data frame  
 LONG\_VR\_DATA <-  
 rbind(LONG\_VR\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
   
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(VR\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$congruent\_and\_audio\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <- participant\_data$Correct\_answers\_congruent\_and\_audio  
 # define the stimuli type  
 stimuli\_type <- "audio"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "congruent"  
 # add the new row to the data frame  
 LONG\_VR\_DATA <-  
 rbind(LONG\_VR\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
   
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(VR\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$congruent\_and\_visual\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <- participant\_data$Correct\_answers\_congruent\_and\_visual  
 # define the stimuli type  
 stimuli\_type <- "visual"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "congruent"  
 # add the new row to the data frame  
 LONG\_VR\_DATA <-  
 rbind(LONG\_VR\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
   
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(VR\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$congruent\_and\_audio.visual\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <-  
 participant\_data$Correct\_answers\_congruent\_and\_audio.visual  
 # define the stimuli type  
 stimuli\_type <- "audio.visual"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "congruent"  
 # add the new row to the data frame  
 LONG\_VR\_DATA <-  
 rbind(LONG\_VR\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
   
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(VR\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$neutral\_and\_audio\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <- participant\_data$Correct\_answers\_neutral\_and\_audio  
 # define the stimuli type  
 stimuli\_type <- "audio"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "neutral"  
 # add the new row to the data frame  
 LONG\_VR\_DATA <-  
 rbind(LONG\_VR\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
   
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(VR\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$neutral\_and\_visual\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <- participant\_data$Correct\_answers\_neutral\_and\_visual  
 # define the stimuli type  
 stimuli\_type <- "visual"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "neutral"  
 # add the new row to the data frame  
 LONG\_VR\_DATA <-  
 rbind(LONG\_VR\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
   
 # filter the data frame to get the rows for the current participant  
 participant\_data <- filter(VR\_not\_done, id == unique\_ID[i])  
 # extract RT  
 RT <- participant\_data$neutral\_and\_audio.visual\_RT  
 # define the ID  
 ID <- participant\_data$id  
 # extract corrects  
 corrects <-  
 participant\_data$Correct\_answers\_neutral\_and\_audio.visual  
 # define the stimuli type  
 stimuli\_type <- "audio.visual"  
 # define the stimuli compatibility  
 stimuli\_compatibility <- "neutral"  
 # add the new row to the data frame  
 LONG\_VR\_DATA <-  
 rbind(LONG\_VR\_DATA,  
 data.frame(ID, stimuli\_type, stimuli\_compatibility, RT, corrects))  
}  
# putting the two data sets together.  
ATTENTION <- rbind(LONG\_VR\_DATA, LONG\_PC\_DATA)  
# we need to ensure we can tell the environments apart, therefor we need to ad PC & VR  
ATTENTION$condition <- rep(c("VR", "PC"), each = 657)

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_PC\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#  
setwd(  
 "/Users/rasmus/Desktop/Playore VR data/mellemskridt/Attention\_data/PC attention data RAW"  
)  
# Instruct R to look in the directory and search for CSV files.  
list <- list.files(pattern = ".csv", recursive = TRUE)  
# "i" represents the current participant and is set to loop through all files in the directory.  
# It is set to loop from 1 to the length of the list of files in our directory.  
i <- 1:length(list)  
  
all\_trials = 144  
  
# creating a empty data frame, that can hold the grand performance data.  
PC\_df\_grandperformance <- data\_frame(  
 id = character(),  
 corrects = numeric(),  
 RT = numeric(),  
 acc = numeric(),  
 no\_answerd = numeric(),  
 total\_answerd = numeric(),  
 stringsAsFactors = FALSE  
)  
  
# settup import and extract loop  
for (i in list)  
{  
 myData <- read.csv2(i)  
   
 corrects\_answers\_subset <-  
 subset(myData, myData$Answer.Codes.Block1 == "1")  
 # we need the converte\_number funktion to  
 RT <-  
 mean(convert\_number(corrects\_answers\_subset$ReactionTime.Block1))  
 corrects <- sum(corrects\_answers\_subset$Answer.Codes.Block1 == "1")  
   
 # Here it is importen not to use the correct answers subset. because we need their no answers  
 no\_answerd <- sum(myData$Answer.Codes.Block1 == "0")  
 # calculate participants total answered trials, to get the participants accuracy on their answered trials.  
 total\_answerd <- all\_trials - no\_answerd  
 # accuracy on total asnwerd trials.  
 acc <- (corrects / total\_answerd) \* 100  
 # create af dataframe  
 PC\_df\_grandperformance <-  
 rbind(  
 PC\_df\_grandperformance,  
 data.frame(i, corrects, RT, acc, no\_answerd, total\_answerd)  
 )  
}  
# replace the file names with ID the participants ID numbers  
ID <- c(10103, 10106, 10107, 10108, 10110, 10113, 10114, 10116, 10117, 10123, 10124, 10126, 10127, 10128, 10129, 10131, 10133, 10134, 10136, 10137, 10138, 10139, 10153, 10154, 10156, 10157, 10158, 20002, 20003, 20005, 20006, 20008, 20009, 20020, 20021, 20022, 30002, 30003, 30004, 30006, 30009, 30010, 30012, 30013, 30019, 30020, 30022, 30023, 30024, 30025, 30026, 30027, 30028, 30030, 30032, 30033, 30034, 30035, 30036, 30040, 30042, 30044, 30046, 30048, 30049, 30051, 30052, 30053, 30054, 30055, 30058, 30060, 30061)  
PC\_df\_grandperformance <- mutate(PC\_df\_grandperformance, i = ID)  
# subset the data based on Performance values below 55%  
sub\_data <-  
 PC\_df\_grandperformance[PC\_df\_grandperformance$acc <= 55, ]  
# extract the ID column values and store them in a list  
PC\_bellow\_55Percentage <- sub\_data$i  
  
#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_VR\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#  
setwd(  
 "/Users/rasmus/Desktop/Playore VR data/mellemskridt/Attention\_data/VR attention data RAW"  
)  
# Instruct R to look in the directory and search for CSV files.  
list <- list.files(pattern = ".csv", recursive = TRUE)  
# "i" represents the current participant and is set to loop through all files in the directory.  
# It is set to loop from 1 to the length of the list of files in our directory.  
i <- 1:length(list)  
  
all\_trials = 144  
  
# creating a empty data frame, that can hold the grand performance data.  
VR\_df\_grandperformance <- data\_frame(  
 id = character(),  
 corrects = numeric(),  
 RT = numeric(),  
 acc = numeric(),  
 no\_answerd = numeric(),  
 total\_answerd = numeric(),  
 stringsAsFactors = FALSE  
)  
  
# settup import and extract loop  
for (i in list)  
{  
 myData <- read.csv2(i)  
   
 corrects\_answers\_subset <-  
 subset(myData, myData$Answer.Codes.Block1 == "1")  
 # we need the converte\_number funktion to  
 RT <-  
 mean(convert\_number(corrects\_answers\_subset$ReactionTime.Block1))  
 corrects <- sum(corrects\_answers\_subset$Answer.Codes.Block1 == "1")  
   
 # Here it is importen not to use the correct answers subset. because we need their no answers  
 no\_answerd <- sum(myData$Answer.Codes.Block1 == "0")  
 # calculate participants total answered trials, to get the participants accuracy on their answered trials.  
 total\_answerd <- all\_trials - no\_answerd  
 # accuracy on total asnwerd trials.  
 acc <- (corrects / total\_answerd) \* 100  
 # create af dataframe  
 VR\_df\_grandperformance <-  
 rbind(  
 VR\_df\_grandperformance,  
 data.frame(i, corrects, RT, acc, no\_answerd, total\_answerd)  
 )  
}  
# replace the file names with ID the participants ID numbers  
ID <- c(10103, 10106, 10107, 10108, 10110, 10113, 10114, 10116, 10117, 10123, 10124, 10126, 10127, 10128, 10129, 10131, 10133, 10134, 10136, 10137, 10138, 10139, 10153, 10154, 10156, 10157, 10158, 20002, 20003, 20005, 20006, 20008, 20009, 20020, 20021, 20022, 30002, 30003, 30004, 30006, 30009, 30010, 30012, 30013, 30019, 30020, 30022, 30023, 30024, 30025, 30026, 30027, 30028, 30030, 30032, 30033, 30034, 30035, 30036, 30040, 30042, 30044, 30046, 30048, 30049, 30051, 30052, 30053, 30054, 30055, 30058, 30060, 30061)  
VR\_df\_grandperformance <- mutate(VR\_df\_grandperformance, i = ID)  
# subset the data based on Performance values below 55%  
sub\_data <-  
 VR\_df\_grandperformance[VR\_df\_grandperformance$acc <= 55, ]  
# extract the ID column values and store them in a list  
VR\_bellow\_55Percentage <- sub\_data$i

# Now that we have to lists with the participants who did not perform above chance lvl, we can extract them from the complete data set.  
PC\_bellow\_55Percentage

## [1] 10108 10113 10124 10126 10131 10134 10136 10154 20003 20020 20022 30002  
## [13] 30003 30004 30006 30009 30022 30023 30026 30032 30044 30060 30061

VR\_bellow\_55Percentage

## [1] 10106 10114 10116 10117 10123 10124 10126 10127 10128 10129 10131 10134  
## [13] 10136 20003 20006 20020 20021 20022 30002 30004 30006 30012 30013 30022  
## [25] 30023 30026 30030 30032 30040 30044 30048 30054 30061

# find the common IDs in both lists  
common\_IDs <-  
 intersect(PC\_bellow\_55Percentage, VR\_bellow\_55Percentage)  
# find the IDs that are only in the PC\_bellow\_55Percentage list  
PC\_only <- setdiff(PC\_bellow\_55Percentage, VR\_bellow\_55Percentage)  
# find the IDs that are only in the VR\_bellow\_55Percentage list  
VR\_only <- setdiff(VR\_bellow\_55Percentage, PC\_bellow\_55Percentage)  
# print the results  
cat("The number of identical IDs in both lists is:", length(common\_IDs))

## The number of identical IDs in both lists is: 17

cat("The common IDs are:", common\_IDs)

## The common IDs are: 10124 10126 10131 10134 10136 20003 20020 20022 30002 30004 30006 30022 30023 30026 30032 30044 30061

cat("The number of IDs only in the PC\_bellow\_55Percentage list is:",  
 length(PC\_only))

## The number of IDs only in the PC\_bellow\_55Percentage list is: 6

cat("The IDs only in the PC\_bellow\_55Percentage list are:", PC\_only)

## The IDs only in the PC\_bellow\_55Percentage list are: 10108 10113 10154 30003 30009 30060

cat("The number of IDs only in the VR\_bellow\_55Percentage list is:",  
 length(VR\_only))

## The number of IDs only in the VR\_bellow\_55Percentage list is: 16

cat("The IDs only in the VR\_bellow\_55Percentage list are:", VR\_only)

## The IDs only in the VR\_bellow\_55Percentage list are: 10106 10114 10116 10117 10123 10127 10128 10129 20006 20021 30012 30013 30030 30040 30048 30054

# 39 children needs to be removed from the complete data set.  
cihldren\_who\_did\_not\_meet\_55percentage = length(common\_IDs) + length(PC\_only) +  
 length(VR\_only)  
  
# Great! We have obtained quality results from our analysis. As a next step, we can calculate the percentage of children that did not meet the 55% criteria in each condition. This will help us understand the impact of the experimental conditions on participant performance and identify any potential limitations of the study.  
length(PC\_only) / cihldren\_who\_did\_not\_meet\_55percentage \* 100 # percentage of children who did not perform over 55% in the PC condition

## [1] 15.38462

length(VR\_only) / cihldren\_who\_did\_not\_meet\_55percentage \* 100 # percentage of children who did not perform over 55% in the VR condition

## [1] 41.02564

length(common\_IDs) / cihldren\_who\_did\_not\_meet\_55percentage \* 100 # percentage of children who did not perform over 55% in both condition

## [1] 43.58974

# removal of children bellow 55% accuracy.  
ATTENTION\_filtered <- ATTENTION  
ATTENTION\_filtered <-  
 filter(ATTENTION\_filtered,!ID %in% common\_IDs)  
ATTENTION\_filtered <- filter(ATTENTION\_filtered,!ID %in% PC\_only)  
ATTENTION\_filtered <- filter(ATTENTION\_filtered,!ID %in% VR\_only)  
  
  
# Bind VR\_df\_grandperformance and PC\_df\_grandperformance together  
combined\_df <- rbind(VR\_df\_grandperformance, PC\_df\_grandperformance)  
# adding a column with the condition label.  
combined\_df$condition <- rep(c("VR", "PC"), each = 73)  
# remove any participants who did not meet the 55% criteria before proceeding with the later step of removing outliers. This will ensure that we are only removing outliers from a reliable and consistent data set.  
combined\_df\_filtered <- combined\_df  
combined\_df\_filtered <-  
 filter(combined\_df\_filtered,!i %in% common\_IDs)  
combined\_df\_filtered <-  
 filter(combined\_df\_filtered,!i %in% PC\_only)  
combined\_df\_filtered <-  
 filter(combined\_df\_filtered,!i %in% VR\_only)  
  
  
# We are now calculating the Z-scores for each participant, which will represent the number of standard deviations (SD) that each participant is from the mean. This will allow us to compare participants on a common scale and make more accurate statistical inferences.  
combined\_df\_filtered <- combined\_df\_filtered %>%  
 mutate(  
 zscore = (  
 combined\_df\_filtered$corrects - mean(combined\_df\_filtered$corrects, na.rm =  
 T)  
 ) / sd(combined\_df\_filtered$corrects, na.rm = T)  
 )  
ID <- combined\_df\_filtered$i  
group <- combined\_df\_filtered$condition  
z <- combined\_df\_filtered$zscore  
performance <- combined\_df\_filtered$corrects  
data.frame(ID, group, performance, z)

## ID group performance z  
## 1 10103 VR 112 0.622933070  
## 2 10107 VR 128 1.377667125  
## 3 10110 VR 67 -1.499756457  
## 4 10133 VR 117 0.858787462  
## 5 10137 VR 86 -0.603509768  
## 6 10138 VR 111 0.575762192  
## 7 10139 VR 99 0.009711651  
## 8 10153 VR 87 -0.556338889  
## 9 10156 VR 115 0.764445706  
## 10 10157 VR 74 -1.169560308  
## 11 10158 VR 93 -0.273313619  
## 12 20002 VR 48 -2.396003146  
## 13 20005 VR 109 0.481420435  
## 14 20008 VR 113 0.670103949  
## 15 20009 VR 93 -0.273313619  
## 16 30010 VR 95 -0.178971862  
## 17 30019 VR 90 -0.414826254  
## 18 30020 VR 138 1.849375909  
## 19 30024 VR 102 0.151224287  
## 20 30025 VR 99 0.009711651  
## 21 30027 VR 103 0.198395165  
## 22 30028 VR 110 0.528591314  
## 23 30033 VR 87 -0.556338889  
## 24 30034 VR 80 -0.886535038  
## 25 30035 VR 33 -3.103566322  
## 26 30036 VR 93 -0.273313619  
## 27 30042 VR 83 -0.745022403  
## 28 30046 VR 60 -1.829952606  
## 29 30049 VR 113 0.670103949  
## 30 30051 VR 98 -0.037459227  
## 31 30052 VR 86 -0.603509768  
## 32 30053 VR 126 1.283325368  
## 33 30055 VR 82 -0.792193281  
## 34 30058 VR 96 -0.131800984  
## 35 10103 PC 72 -1.263902065  
## 36 10107 PC 124 1.188983611  
## 37 10110 PC 115 0.764445706  
## 38 10133 PC 127 1.330496246  
## 39 10137 PC 84 -0.697851524  
## 40 10138 PC 106 0.339907800  
## 41 10139 PC 85 -0.650680646  
## 42 10153 PC 105 0.292736922  
## 43 10156 PC 131 1.519179760  
## 44 10157 PC 122 1.094641854  
## 45 10158 PC 124 1.188983611  
## 46 20002 PC 56 -2.018636119  
## 47 20005 PC 87 -0.556338889  
## 48 20008 PC 94 -0.226142741  
## 49 20009 PC 89 -0.461997132  
## 50 30010 PC 119 0.953129219  
## 51 30019 PC 84 -0.697851524  
## 52 30020 PC 131 1.519179760  
## 53 30024 PC 89 -0.461997132  
## 54 30025 PC 128 1.377667125  
## 55 30027 PC 128 1.377667125  
## 56 30028 PC 120 1.000300098  
## 57 30033 PC 103 0.198395165  
## 58 30034 PC 122 1.094641854  
## 59 30035 PC 91 -0.367655376  
## 60 30036 PC 93 -0.273313619  
## 61 30042 PC 88 -0.509168011  
## 62 30046 PC 73 -1.216731187  
## 63 30049 PC 117 0.858787462  
## 64 30051 PC 106 0.339907800  
## 65 30052 PC 80 -0.886535038  
## 66 30053 PC 112 0.622933070  
## 67 30055 PC 102 0.151224287  
## 68 30058 PC 85 -0.650680646

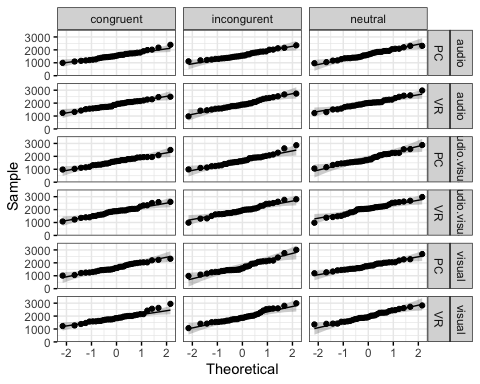
combined\_df\_filtered <- combined\_df\_filtered %>%  
 mutate(zscore = (  
 combined\_df\_filtered$RT - mean(combined\_df\_filtered$RT, na.rm = T)  
 ) / sd(combined\_df\_filtered$RT, na.rm = T))  
ID <- combined\_df\_filtered$i  
group <- combined\_df\_filtered$condition  
z <- combined\_df\_filtered$zscore  
performance <- combined\_df\_filtered$RT  
data.frame(ID, group, performance, z)

## ID group performance z  
## 1 10103 VR 2.1885443 1.112803210  
## 2 10107 VR 1.6552517 -0.376604618  
## 3 10110 VR 1.8400681 0.139560482  
## 4 10133 VR 1.9040593 0.318278531  
## 5 10137 VR 1.9506254 0.448330613  
## 6 10138 VR 2.4866543 1.945380325  
## 7 10139 VR 1.7960758 0.016696446  
## 8 10153 VR 2.4611011 1.874014065  
## 9 10156 VR 1.8809774 0.253814253  
## 10 10157 VR 2.4389835 1.812243036  
## 11 10158 VR 2.1263956 0.939230859  
## 12 20002 VR 2.1006263 0.867261047  
## 13 20005 VR 2.3979544 1.697654729  
## 14 20008 VR 2.2036677 1.155040574  
## 15 20009 VR 1.9932670 0.567422524  
## 16 30010 VR 1.7399162 -0.140149020  
## 17 30019 VR 1.8063576 0.045411911  
## 18 30020 VR 1.5958930 -0.542384700  
## 19 30024 VR 1.7916802 0.004420186  
## 20 30025 VR 1.4160484 -1.044664018  
## 21 30027 VR 1.6952396 -0.264924300  
## 22 30028 VR 2.4045195 1.715990050  
## 23 30033 VR 2.0130397 0.622644686  
## 24 30034 VR 1.8891054 0.276514423  
## 25 30035 VR 1.6806755 -0.305599535  
## 26 30036 VR 1.2144668 -1.607651630  
## 27 30042 VR 1.4314205 -1.001731995  
## 28 30046 VR 2.0242134 0.653851172  
## 29 30049 VR 2.0553521 0.740817071  
## 30 30051 VR 2.1495354 1.003857150  
## 31 30052 VR 1.5478345 -0.676604902  
## 32 30053 VR 1.9126600 0.342299096  
## 33 30055 VR 1.9828366 0.538291921  
## 34 30058 VR 2.1624798 1.040008890  
## 35 10103 PC 1.8940179 0.290234206  
## 36 10107 PC 1.4080111 -1.067110951  
## 37 10110 PC 1.6002823 -0.530125995  
## 38 10133 PC 1.5911416 -0.555654452  
## 39 10137 PC 1.8906090 0.280713737  
## 40 10138 PC 2.4233784 1.768660115  
## 41 10139 PC 1.7906883 0.001650095  
## 42 10153 PC 1.4450486 -0.963670709  
## 43 10156 PC 1.3994769 -1.090945902  
## 44 10157 PC 1.9747085 0.515591242  
## 45 10158 PC 2.1548393 1.018670071  
## 46 20002 PC 1.7415744 -0.135517766  
## 47 20005 PC 1.7879593 -0.005971628  
## 48 20008 PC 1.8537143 0.177672331  
## 49 20009 PC 1.4850604 -0.851923766  
## 50 30010 PC 1.4308618 -1.003292487  
## 51 30019 PC 1.2553347 -1.493513724  
## 52 30020 PC 1.6416393 -0.414621982  
## 53 30024 PC 1.3036694 -1.358521937  
## 54 30025 PC 1.0791214 -1.985651558  
## 55 30027 PC 1.4219663 -1.028136209  
## 56 30028 PC 1.8014323 0.031656345  
## 57 30033 PC 1.9743267 0.514524829  
## 58 30034 PC 1.9219165 0.368151118  
## 59 30035 PC 0.7666611 -2.858307006  
## 60 30036 PC 1.5633432 -0.633291294  
## 61 30042 PC 1.2568797 -1.489198617  
## 62 30046 PC 1.5469531 -0.679066379  
## 63 30049 PC 2.1608140 1.035356509  
## 64 30051 PC 2.0433103 0.707186039  
## 65 30052 PC 1.3503354 -1.228190751  
## 66 30053 PC 1.7358165 -0.151598842  
## 67 30055 PC 1.7783584 -0.032785679  
## 68 30058 PC 1.3158542 -1.324491532

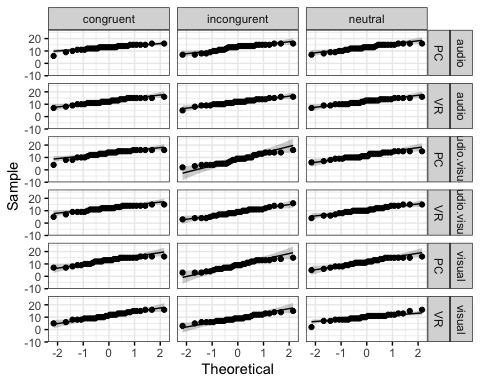
# We are now removing any participants that have outliers that exceed more than 2 standard deviations from the mean. This is an important step to ensure that our data set is not skewed by extreme values and that our analysis is based on reliable data.  
ATTENTION\_filtered <- filter(ATTENTION\_filtered, ID != "20002")  
ATTENTION\_filtered <- filter(ATTENTION\_filtered, ID != "30035")  
  
# We are now setting the complete data set by removing any outliers and participants who did not meet the 55% criteria. This will ensure that our analysis is based on a clean and reliable data set.  
ATTENTION <- ATTENTION\_filtered  
# Convert seconds to milliseconds  
ATTENTION$RT <- ATTENTION$RT \* 1000  
# remove all objects except for mydata and mylist  
rm(list = setdiff(ls(), c("ATTENTION")))  
# We have a total of 32 participants, and none of them have outliers that exceed more than 2 standard deviations from the mean. Additionally, all participants meet the 55% criteria.  
ATTENTION %>%  
 group\_by(condition) %>%  
 summarize(distinct\_points = n\_distinct(ID))

## # A tibble: 2 × 2  
## condition distinct\_points  
## <chr> <int>  
## 1 PC 32  
## 2 VR 32

# We are plotting the quantile-quantile plots to ensure normal distribution of the data. We decided not to use the Shapiro-Wilk test due to its sensitivity to large sample sizes.  
ggqqplot(ATTENTION, "RT", ggtheme = theme\_bw()) +  
 facet\_grid(stimuli\_type ~ condition ~ stimuli\_compatibility)



ggqqplot(ATTENTION, "corrects", ggtheme = theme\_bw()) +  
 facet\_grid(stimuli\_type ~ condition ~ stimuli\_compatibility)



# We are using the Levene's test to check for differences in variance between the two groups. This is to ensure that the assumption of homogeneity of variance is met.  
ATTENTION %>%  
 group\_by(stimuli\_compatibility , stimuli\_type) %>%  
 levene\_test(RT ~ condition)

## # A tibble: 9 × 6  
## stimuli\_type stimuli\_compatibility df1 df2 statistic p  
## <chr> <chr> <int> <int> <dbl> <dbl>  
## 1 audio congruent 1 62 0.136 0.713  
## 2 audio.visual congruent 1 62 0.174 0.678  
## 3 visual congruent 1 62 0.0231 0.880  
## 4 audio incongurent 1 62 1.05 0.310  
## 5 audio.visual incongurent 1 62 0.00000341 0.999  
## 6 visual incongurent 1 62 0.115 0.736  
## 7 audio neutral 1 62 0.192 0.663  
## 8 audio.visual neutral 1 62 0.0927 0.762  
## 9 visual neutral 1 62 1.83 0.182

ATTENTION %>%  
 group\_by(stimuli\_compatibility , stimuli\_type) %>%  
 levene\_test(corrects ~ condition)

## # A tibble: 9 × 6  
## stimuli\_type stimuli\_compatibility df1 df2 statistic p  
## <chr> <chr> <int> <int> <dbl> <dbl>  
## 1 audio congruent 1 62 1.34 0.251  
## 2 audio.visual congruent 1 62 0.551 0.461  
## 3 visual congruent 1 62 0.616 0.435  
## 4 audio incongurent 1 62 0.192 0.663  
## 5 audio.visual incongurent 1 62 1.88 0.176  
## 6 visual incongurent 1 62 2.50 0.119  
## 7 audio neutral 1 62 0.182 0.671  
## 8 audio.visual neutral 1 62 0.00592 0.939  
## 9 visual neutral 1 62 1.56 0.217

# In the first few lines, we are creating a new data frame called "PC" that is a subset of the original "ATTENTION" data frame. We are selecting only the rows where the "condition" column equals "PC".  
PC <- subset(ATTENTION, condition == "PC")  
  
# We are then calculating the means, standard deviations, and sample sizes for RT and corrects for each level of "stimuli\_type" using the dplyr function "get\_summary\_stats". This is done separately for each variable.  
PC %>%  
 group\_by(stimuli\_type) %>%  
 get\_summary\_stats(RT, type = "mean\_sd")

## # A tibble: 3 × 5  
## stimuli\_type variable n mean sd  
## <chr> <chr> <dbl> <dbl> <dbl>  
## 1 audio RT 96 1609. 328.  
## 2 audio.visual RT 96 1713. 403.  
## 3 visual RT 96 1713. 389.

PC %>%  
 group\_by(stimuli\_type) %>%  
 get\_summary\_stats(corrects, type = "mean\_sd")

## # A tibble: 3 × 5  
## stimuli\_type variable n mean sd  
## <chr> <chr> <dbl> <dbl> <dbl>  
## 1 audio corrects 96 12.8 2.39  
## 2 audio.visual corrects 96 11.0 3.65  
## 3 visual corrects 96 11.1 3.47

# We then repeat the above step, but grouping by "stimuli\_compatibility" instead of "stimuli\_type".  
PC %>%  
 group\_by(stimuli\_compatibility) %>%  
 get\_summary\_stats(corrects, type = "mean\_sd")

## # A tibble: 3 × 5  
## stimuli\_compatibility variable n mean sd  
## <chr> <chr> <dbl> <dbl> <dbl>  
## 1 congruent corrects 96 12.9 2.54  
## 2 incongurent corrects 96 10.0 3.88  
## 3 neutral corrects 96 12.0 2.69

PC %>%  
 group\_by(stimuli\_compatibility) %>%  
 get\_summary\_stats(RT, type = "mean\_sd")

## # A tibble: 3 × 5  
## stimuli\_compatibility variable n mean sd  
## <chr> <chr> <dbl> <dbl> <dbl>  
## 1 congruent RT 96 1604. 332.  
## 2 incongurent RT 96 1678. 407.  
## 3 neutral RT 96 1753. 377.

# Next, we calculate the means, standard deviations, and sample sizes for RT and corrects for each combination of "stimuli\_type" and "stimuli\_compatibility" using "get\_summary\_stats".  
PC %>%  
 group\_by(stimuli\_type, stimuli\_compatibility) %>%  
 get\_summary\_stats(corrects, type = "mean\_sd")

## # A tibble: 9 × 6  
## stimuli\_type stimuli\_compatibility variable n mean sd  
## <chr> <chr> <chr> <dbl> <dbl> <dbl>  
## 1 audio congruent corrects 32 12.9 2.08  
## 2 audio incongurent corrects 32 12.6 2.66  
## 3 audio neutral corrects 32 12.8 2.46  
## 4 audio.visual congruent corrects 32 13.0 2.83  
## 5 audio.visual incongurent corrects 32 8.5 3.91  
## 6 audio.visual neutral corrects 32 11.6 2.52  
## 7 visual congruent corrects 32 12.7 2.71  
## 8 visual incongurent corrects 32 8.94 3.64  
## 9 visual neutral corrects 32 11.6 2.97

PC %>%  
 group\_by(stimuli\_type, stimuli\_compatibility) %>%  
 get\_summary\_stats(RT, type = "mean\_sd")

## # A tibble: 9 × 6  
## stimuli\_type stimuli\_compatibility variable n mean sd  
## <chr> <chr> <chr> <dbl> <dbl> <dbl>  
## 1 audio congruent RT 32 1566. 317.  
## 2 audio incongurent RT 32 1611. 312.  
## 3 audio neutral RT 32 1651. 360.  
## 4 audio.visual congruent RT 32 1602. 336.  
## 5 audio.visual incongurent RT 32 1702. 423.  
## 6 audio.visual neutral RT 32 1835. 423.  
## 7 visual congruent RT 32 1644. 348.  
## 8 visual incongurent RT 32 1721. 474.  
## 9 visual neutral RT 32 1775. 330.

# Finally, we perform paired-t-tests to compare the means of corrects and RT between each level of "stimuli\_compatibility" for each level of "stimuli\_type". We use the "t\_test" function from the "rstatix" package, and adjust the p-values using the Bonferroni correction.  
PC %>%  
 group\_by(stimuli\_type) %>%  
 t\_test(corrects ~ stimuli\_compatibility,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 9 × 11  
## stimul…¹ .y. group1 group2 n1 n2 stati…² df p p.adj p.adj…³  
## \* <chr> <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl> <dbl> <chr>   
## 1 audio corr… congr… incon… 32 32 0.590 31 5.59e-1 1 e+0 ns   
## 2 audio corr… congr… neutr… 32 32 0.296 31 7.7 e-1 1 e+0 ns   
## 3 audio corr… incon… neutr… 32 32 -0.239 31 8.13e-1 1 e+0 ns   
## 4 audio.v… corr… congr… incon… 32 32 5.95 31 1.43e-6 4.29e-6 \*\*\*\*   
## 5 audio.v… corr… congr… neutr… 32 32 3.03 31 5 e-3 1.5 e-2 \*   
## 6 audio.v… corr… incon… neutr… 32 32 -5.29 31 9.23e-6 2.77e-5 \*\*\*\*   
## 7 visual corr… congr… incon… 32 32 5.46 31 5.77e-6 1.73e-5 \*\*\*\*   
## 8 visual corr… congr… neutr… 32 32 2.13 31 4.1 e-2 1.23e-1 ns   
## 9 visual corr… incon… neutr… 32 32 -4.84 31 3.37e-5 1.01e-4 \*\*\*   
## # … with abbreviated variable names ¹​stimuli\_type, ²​statistic, ³​p.adj.signif

PC %>%  
 group\_by(stimuli\_type) %>%  
 t\_test(RT ~ stimuli\_compatibility,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 9 × 11  
## stimul…¹ .y. group1 group2 n1 n2 stati…² df p p.adj p.adj…³  
## \* <chr> <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl> <dbl> <chr>   
## 1 audio RT congr… incon… 32 32 -1.21 31 2.35e-1 7.05e-1 ns   
## 2 audio RT congr… neutr… 32 32 -2.41 31 2.2 e-2 6.6 e-2 ns   
## 3 audio RT incon… neutr… 32 32 -1.15 31 2.57e-1 7.71e-1 ns   
## 4 audio.v… RT congr… incon… 32 32 -1.60 31 1.19e-1 3.57e-1 ns   
## 5 audio.v… RT congr… neutr… 32 32 -4.12 31 2.62e-4 7.86e-4 \*\*\*   
## 6 audio.v… RT incon… neutr… 32 32 -1.68 31 1.03e-1 3.09e-1 ns   
## 7 visual RT congr… incon… 32 32 -1.16 31 2.57e-1 7.71e-1 ns   
## 8 visual RT congr… neutr… 32 32 -3.29 31 3 e-3 8 e-3 \*\*   
## 9 visual RT incon… neutr… 32 32 -0.863 31 3.95e-1 1 e+0 ns   
## # … with abbreviated variable names ¹​stimuli\_type, ²​statistic, ³​p.adj.signif

# Additionally, we perform paired-t-tests to compare the means of corrects and RT between each level of "stimuli\_type" for each level of "stimuli\_compatibility". These are also adjusted using the Bonferroni correction.  
PC %>%  
 t\_test(corrects ~ stimuli\_type,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 3 × 10  
## .y. group1 group2 n1 n2 stati…¹ df p p.adj p.adj…²  
## \* <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl> <dbl> <chr>   
## 1 corrects audio audio… 96 96 4.95 95 3.2 e-6 9.6 e-6 \*\*\*\*   
## 2 corrects audio visual 96 96 5.01 95 2.48e-6 7.44e-6 \*\*\*\*   
## 3 corrects audio.visual visual 96 96 -0.0400 95 9.68e-1 1 e+0 ns   
## # … with abbreviated variable names ¹​statistic, ²​p.adj.signif

PC %>%  
 t\_test(corrects ~ stimuli\_compatibility,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 3 × 10  
## .y. group1 group2 n1 n2 stati…¹ df p p.adj p.adj…²  
## \* <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl> <dbl> <chr>   
## 1 corrects congruent incon… 96 96 6.88 95 6.25e-10 1.88e-9 \*\*\*\*   
## 2 corrects congruent neutr… 96 96 3.21 95 2 e- 3 5 e-3 \*\*   
## 3 corrects incongurent neutr… 96 96 -5.70 95 1.34e- 7 4.02e-7 \*\*\*\*   
## # … with abbreviated variable names ¹​statistic, ²​p.adj.signif

PC %>%  
 t\_test(RT ~ stimuli\_type,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 3 × 10  
## .y. group1 group2 n1 n2 stati…¹ df p p.adj p.adj…²  
## \* <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl> <dbl> <chr>   
## 1 RT audio audio.vi… 96 96 -3.58 95 5.42e-4 2 e-3 \*\*   
## 2 RT audio visual 96 96 -4.13 95 7.75e-5 2.32e-4 \*\*\*   
## 3 RT audio.visual visual 96 96 -0.0193 95 9.85e-1 1 e+0 ns   
## # … with abbreviated variable names ¹​statistic, ²​p.adj.signif

PC %>%  
 t\_test(RT ~ stimuli\_compatibility,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 3 × 10  
## .y. group1 group2 n1 n2 stati…¹ df p p.adj p.adj…²  
## \* <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl> <dbl> <chr>   
## 1 RT congruent incongure… 96 96 -2.27 95 2.5 e-2 7.6 e-2 ns   
## 2 RT congruent neutral 96 96 -5.67 95 1.53e-7 4.59e-7 \*\*\*\*   
## 3 RT incongurent neutral 96 96 -2.13 95 3.5 e-2 1.06e-1 ns   
## # … with abbreviated variable names ¹​statistic, ²​p.adj.signif

# In the first few lines, we are creating a new data frame called "VR" that is a subset of the original "ATTENTION" data frame. We are selecting only the rows where the "condition" column equals "VR".  
VR <- subset(ATTENTION, condition == "VR")  
  
# We are then calculating the means, standard deviations, and sample sizes for RT and corrects for each level of "stimuli\_type" using the dplyr function "get\_summary\_stats". This is done separately for each variable.  
VR %>%  
 group\_by(stimuli\_type) %>%  
 get\_summary\_stats(RT, type = "mean\_sd")

## # A tibble: 3 × 5  
## stimuli\_type variable n mean sd  
## <chr> <chr> <dbl> <dbl> <dbl>  
## 1 audio RT 96 1934. 364.  
## 2 audio.visual RT 96 1957. 414.  
## 3 visual RT 96 1943. 412.

VR %>%  
 group\_by(stimuli\_type) %>%  
 get\_summary\_stats(corrects, type = "mean\_sd")

## # A tibble: 3 × 5  
## stimuli\_type variable n mean sd  
## <chr> <chr> <dbl> <dbl> <dbl>  
## 1 audio corrects 96 12.1 2.28  
## 2 audio.visual corrects 96 10.4 3.01  
## 3 visual corrects 96 10.2 2.93

# We then repeat the above step, but grouping by "stimuli\_compatibility" instead of "stimuli\_type".  
VR %>%  
 group\_by(stimuli\_compatibility) %>%  
 get\_summary\_stats(corrects, type = "mean\_sd")

## # A tibble: 3 × 5  
## stimuli\_compatibility variable n mean sd  
## <chr> <chr> <dbl> <dbl> <dbl>  
## 1 congruent corrects 96 11.8 2.54  
## 2 incongurent corrects 96 9.95 3.12  
## 3 neutral corrects 96 11.0 2.65

VR %>%  
 group\_by(stimuli\_compatibility) %>%  
 get\_summary\_stats(RT, type = "mean\_sd")

## # A tibble: 3 × 5  
## stimuli\_compatibility variable n mean sd  
## <chr> <chr> <dbl> <dbl> <dbl>  
## 1 congruent RT 96 1874. 356.  
## 2 incongurent RT 96 1945. 419.  
## 3 neutral RT 96 2015. 403.

# Next, we calculate the means, standard deviations, and sample sizes for RT and corrects for each combination of "stimuli\_type" and "stimuli\_compatibility" using "get\_summary\_stats".  
VR %>%  
 group\_by(stimuli\_type, stimuli\_compatibility) %>%  
 get\_summary\_stats(corrects, type = "mean\_sd")

## # A tibble: 9 × 6  
## stimuli\_type stimuli\_compatibility variable n mean sd  
## <chr> <chr> <chr> <dbl> <dbl> <dbl>  
## 1 audio congruent corrects 32 12.3 2.27  
## 2 audio incongurent corrects 32 11.8 2.44  
## 3 audio neutral corrects 32 12.2 2.17  
## 4 audio.visual congruent corrects 32 11.8 2.35  
## 5 audio.visual incongurent corrects 32 8.88 3.19  
## 6 audio.visual neutral corrects 32 10.6 2.76  
## 7 visual congruent corrects 32 11.3 2.95  
## 8 visual incongurent corrects 32 9.22 2.94  
## 9 visual neutral corrects 32 10.1 2.58

VR %>%  
 group\_by(stimuli\_type, stimuli\_compatibility) %>%  
 get\_summary\_stats(RT, type = "mean\_sd")

## # A tibble: 9 × 6  
## stimuli\_type stimuli\_compatibility variable n mean sd  
## <chr> <chr> <chr> <dbl> <dbl> <dbl>  
## 1 audio congruent RT 32 1881. 316.  
## 2 audio incongurent RT 32 1927. 398.  
## 3 audio neutral RT 32 1995. 376.  
## 4 audio.visual congruent RT 32 1860. 378.  
## 5 audio.visual incongurent RT 32 1956. 421.  
## 6 audio.visual neutral RT 32 2056. 431.  
## 7 visual congruent RT 32 1881. 381.  
## 8 visual incongurent RT 32 1953. 449.  
## 9 visual neutral RT 32 1994. 409.

# Finally, we perform paired-t-tests to compare the means of corrects and RT between each level of "stimuli\_compatibility" for each level of "stimuli\_type". We use the "t\_test" function from the "rstatix" package, and adjust the p-values using the Bonferroni correction.  
VR %>%  
 group\_by(stimuli\_type) %>%  
 t\_test(corrects ~ stimuli\_compatibility,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 9 × 11  
## stimul…¹ .y. group1 group2 n1 n2 stati…² df p p.adj p.adj…³  
## \* <chr> <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl> <dbl> <chr>   
## 1 audio corr… congr… incon… 32 32 1.30 31 2.04e-1 6.12e-1 ns   
## 2 audio corr… congr… neutr… 32 32 0.387 31 7.01e-1 1 e+0 ns   
## 3 audio corr… incon… neutr… 32 32 -1.21 31 2.34e-1 7.02e-1 ns   
## 4 audio.v… corr… congr… incon… 32 32 5.19 31 1.25e-5 3.75e-5 \*\*\*\*   
## 5 audio.v… corr… congr… neutr… 32 32 2.20 31 3.5 e-2 1.06e-1 ns   
## 6 audio.v… corr… incon… neutr… 32 32 -3.94 31 4.35e-4 1 e-3 \*\*   
## 7 visual corr… congr… incon… 32 32 3.96 31 4.07e-4 1 e-3 \*\*   
## 8 visual corr… congr… neutr… 32 32 2.65 31 1.2 e-2 3.7 e-2 \*   
## 9 visual corr… incon… neutr… 32 32 -1.83 31 7.7 e-2 2.32e-1 ns   
## # … with abbreviated variable names ¹​stimuli\_type, ²​statistic, ³​p.adj.signif

VR %>%  
 group\_by(stimuli\_type) %>%  
 t\_test(RT ~ stimuli\_compatibility,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 9 × 11  
## stimuli\_type .y. group1 group2 n1 n2 stati…¹ df p p.adj p.adj…²  
## \* <chr> <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl> <dbl> <chr>   
## 1 audio RT congr… incon… 32 32 -0.849 31 0.403 1 ns   
## 2 audio RT congr… neutr… 32 32 -2.22 31 0.034 0.101 ns   
## 3 audio RT incon… neutr… 32 32 -1.49 31 0.145 0.435 ns   
## 4 audio.visual RT congr… incon… 32 32 -1.68 31 0.103 0.309 ns   
## 5 audio.visual RT congr… neutr… 32 32 -3.55 31 0.001 0.004 \*\*   
## 6 audio.visual RT incon… neutr… 32 32 -1.58 31 0.124 0.372 ns   
## 7 visual RT congr… incon… 32 32 -0.923 31 0.363 1 ns   
## 8 visual RT congr… neutr… 32 32 -1.56 31 0.129 0.387 ns   
## 9 visual RT incon… neutr… 32 32 -0.585 31 0.563 1 ns   
## # … with abbreviated variable names ¹​statistic, ²​p.adj.signif

# Additionally, we perform paired-t-tests to compare the means of corrects and RT between each level of "stimuli\_type" for each level of "stimuli\_compatibility". These are also adjusted using the Bonferroni correction.  
VR %>%  
 t\_test(corrects ~ stimuli\_type,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 3 × 10  
## .y. group1 group2 n1 n2 stati…¹ df p p.adj p.adj…²  
## \* <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl> <dbl> <chr>   
## 1 corrects audio audio… 96 96 5.93 95 4.78e- 8 1.43e-7 \*\*\*\*   
## 2 corrects audio visual 96 96 6.88 95 6.46e-10 1.94e-9 \*\*\*\*   
## 3 corrects audio.visu… visual 96 96 0.824 95 4.12e- 1 1 e+0 ns   
## # … with abbreviated variable names ¹​statistic, ²​p.adj.signif

VR %>%  
 t\_test(corrects ~ stimuli\_compatibility,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 3 × 10  
## .y. group1 group2 n1 n2 stati…¹ df p p.adj p.adj…²  
## \* <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl> <dbl> <chr>   
## 1 corrects congruent incong… 96 96 6.00 95 3.63e-8 1.09e-7 \*\*\*\*   
## 2 corrects congruent neutral 96 96 3.23 95 2 e-3 5 e-3 \*\*   
## 3 corrects incongurent neutral 96 96 -4.02 95 1.15e-4 3.45e-4 \*\*\*   
## # … with abbreviated variable names ¹​statistic, ²​p.adj.signif

VR %>%  
 t\_test(RT ~ stimuli\_type,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 3 × 10  
## .y. group1 group2 n1 n2 statis…¹ df p p.adj p.adj…²  
## \* <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl> <dbl> <chr>   
## 1 RT audio audio.visual 96 96 -0.778 95 0.439 1 ns   
## 2 RT audio visual 96 96 -0.230 95 0.818 1 ns   
## 3 RT audio.visual visual 96 96 0.376 95 0.708 1 ns   
## # … with abbreviated variable names ¹​statistic, ²​p.adj.signif

VR %>%  
 t\_test(RT ~ stimuli\_compatibility,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 3 × 10  
## .y. group1 group2 n1 n2 stati…¹ df p p.adj p.adj…²  
## \* <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl> <dbl> <chr>   
## 1 RT congruent incongure… 96 96 -1.95 95 5.4 e-2 1.63e-1 ns   
## 2 RT congruent neutral 96 96 -4.06 95 9.97e-5 2.99e-4 \*\*\*   
## 3 RT incongurent neutral 96 96 -2.01 95 4.7 e-2 1.41e-1 ns   
## # … with abbreviated variable names ¹​statistic, ²​p.adj.signif

# Create a vector with the desired values  
cb\_values <- rep(c("cost", "cost", "cost", "benefit", "benefit", "benefit", "", "", ""), length.out = nrow(data))  
  
# Add the vector as a new column to the data set  
ATTENTION$cost\_benefit <- cb\_values  
  
ATTENTION <-  
 data.frame(ATTENTION, cost\_ben\_val = rep(NA, nrow(ATTENTION)))  
  
# Get unique IDs  
unique\_ids <- unique(ATTENTION$ID)  
# Iterate over unique IDs  
for (i in unique\_ids) {  
 # Subset data for current ID  
 data <- subset(ATTENTION, ID == i)  
   
 #\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_VR\_\_\_\_\_\_\_\_\_\_\_\_\_\_#  
 # define the cost value, (incongruent - neutral trials) for the same stimuli type (audio)  
 cost <- data[1, "RT"] - data[7, "RT"]  
 # Find the first NA value in the 'cost\_ben\_val' column  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
   
 # Assign the 'cost' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- cost  
   
 # define the cost value, (incongruent - neutral trials) for the same stimuli type (visual)  
 cost <- data[2, "RT"] - data[8, "RT"]  
 # Find the first NA value in the 'cost\_ben\_val' column  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
   
 # Assign the 'cost' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- cost  
   
 # define the cost value, (incongruent - neutral trials) for the same stimuli type (audio.visual)  
 cost <- data[3, "RT"] - data[9, "RT"]  
 # Find the first NA value in the 'cost\_ben\_val' column  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
 # Assign the 'cost' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- cost  
   
   
   
 # define the benefit value, (neutral trials - congruent ) for the same stimuli type (audio)  
 benefit <- data[7, "RT"] - data[4, "RT"]  
 # Find the first NA value in the 'cost\_ben\_val' column  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
   
 # Assign the 'benefit' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- benefit  
   
 # define the benefit value, ( neutral trials - congreunt ) for the same stimuli type (visual)  
 benefit <- data[8, "RT"] - data[5, "RT"]  
 # Find the first NA value in the 'cost\_ben\_val' column  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
   
 # Assign the 'benefit' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- benefit  
   
 # define the benefit value, (neutral trials - congreunt) for the same stimuli type (audio.visual)  
 benefit <- data[9, "RT"] - data[6, "RT"]  
 # Find the first NA value in the 'cost\_ben\_val' column  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
 # Assign the 'benefit' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- benefit  
   
   
 # neutral  
 # sett the next three equal to NA due to neutral - neutral is 0 neutral <- "NA"  
 neutral <- "NEUTRAL"  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
   
 # Assign the 'benefit' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- neutral  
   
 # neutral  
 # sett the next three equal to NA due to neutral - neutral is 0 neutral <- "NA"  
 neutral <- "NEUTRAL"  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
   
 # Assign the 'benefit' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- neutral  
   
 # neutral  
 # sett the next three equal to NA due to neutral - neutral is 0 neutral <- "NA"  
 neutral <- "NEUTRAL"  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
   
 # Assign the 'benefit' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- neutral  
   
   
   
 #\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_PC\_\_\_\_\_\_\_\_\_\_\_\_\_\_#  
 # define the cost value, (incongruent - neutral trials) for the same stimuli type (audio)  
 cost <- data[10, "RT"] - data[16, "RT"]  
 # Find the first NA value in the 'cost\_ben\_val' column  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
   
 # Assign the 'cost' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- cost  
   
 # define the cost value, (incongruent - neutral trials) for the same stimuli type (visual)  
 cost <- data[11, "RT"] - data[17, "RT"]  
 # Find the first NA value in the 'cost\_ben\_val' column  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
   
 # Assign the 'cost' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- cost  
   
 # define the cost value, (incongruent - neutral trials) for the same stimuli type (audio.visual)  
 cost <- data[12, "RT"] - data[18, "RT"]  
 # Find the first NA value in the 'cost\_ben\_val' column  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
 # Assign the 'cost' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- cost  
   
   
 # define the benefit value, (neutral trials - congruent ) for the same stimuli type (audio)  
 benefit <- data[16, "RT"] - data[13, "RT"]  
 # Find the first NA value in the 'cost\_ben\_val' column  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
   
 # Assign the 'benefit' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- benefit  
   
 # define the benefit value, ( neutral trials - congreunt ) for the same stimuli type (visual)  
 benefit <- data[17, "RT"] - data[14, "RT"]  
 # Find the first NA value in the 'cost\_ben\_val' column  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
   
 # Assign the 'benefit' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- benefit  
   
 # define the benefit value, (neutral trials - congreunt) for the same stimuli type (audio.visual)  
 benefit <- data[18, "RT"] - data[15, "RT"]  
 # Find the first NA value in the 'cost\_ben\_val' column  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
 # Assign the 'benefit' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- benefit  
   
 # neutral  
 # sett the next three equal to NA due to neutral - neutral is 0 neutral <- "NA"  
 neutral <- "NEUTRAL"  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
   
 # Assign the 'benefit' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- neutral  
   
 # neutral  
 # sett the next three equal to NA due to neutral - neutral is 0 neutral <- "NA"  
 neutral <- "NEUTRAL"  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
   
 # Assign the 'benefit' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- neutral  
   
 # neutral  
 # sett the next three equal to NA due to neutral - neutral is 0 neutral <- "NA"  
 neutral <- "NEUTRAL"  
 first\_na <- which(is.na(ATTENTION$cost\_ben\_val))[1]  
   
 # Assign the 'benefit' value to the first NA in the 'cost\_ben\_val' column  
 ATTENTION$cost\_ben\_val[first\_na] <- neutral  
   
}  
  
# Convert the 'cost\_ben\_val' column of the 'ATTENTION' dataframe to numeric data type  
ATTENTION$cost\_ben\_val <- as.numeric(ATTENTION$cost\_ben\_val)  
  
# Group the 'ATTENTION' dataframe by 'condition', 'stimuli\_type', and 'cost\_benefit' columns and  
# calculate summary statistics (mean and standard deviation) for the 'cost benefit values' column  
ATTENTION %>%  
 group\_by(condition, stimuli\_type , cost\_benefit) %>%  
 get\_summary\_stats(cost\_ben\_val, type = "mean\_sd")

## # A tibble: 12 × 7  
## stimuli\_type condition cost\_benefit variable n mean sd  
## <chr> <chr> <chr> <chr> <dbl> <dbl> <dbl>  
## 1 audio PC benefit cost\_ben\_val 32 114. 222.  
## 2 audio PC cost cost\_ben\_val 32 -48.8 261.  
## 3 audio.visual PC benefit cost\_ben\_val 32 206. 330.  
## 4 audio.visual PC cost cost\_ben\_val 32 -155. 345.  
## 5 visual PC benefit cost\_ben\_val 32 94.8 358.  
## 6 visual PC cost cost\_ben\_val 32 -49.6 422.  
## 7 audio VR benefit cost\_ben\_val 32 84.4 272.  
## 8 audio VR cost cost\_ben\_val 32 -58.6 189.  
## 9 audio.visual VR benefit cost\_ben\_val 32 222. 302.  
## 10 audio.visual VR cost cost\_ben\_val 32 -77.3 452.  
## 11 visual VR benefit cost\_ben\_val 32 148. 295.  
## 12 visual VR cost cost\_ben\_val 32 -44.4 315.

# remove all objects except for mydata and mylist  
rm(list = setdiff(ls(), c("ATTENTION")))

# group the data by both "cost\_benefit" and "stimuli\_type" and perform a paired t-test on the "cost benefit value"  
# Next, we compare the condition and adjust the p-values using the Bonferroni correction.  
# this was done to find diffrences between the conditions. based on the stimuli type.  
ATTENTION %>%  
 filter(!is.na(cost\_ben\_val)) %>%  
 group\_by(cost\_benefit, stimuli\_type) %>%  
 t\_test(cost\_ben\_val ~ condition ,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 6 × 10  
## stimuli\_type cost\_benefit .y. group1 group2 n1 n2 stati…¹ df p  
## \* <chr> <chr> <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl>  
## 1 audio benefit cost\_… PC VR 32 32 0.453 31 0.654  
## 2 audio.visual benefit cost\_… PC VR 32 32 -0.195 31 0.847  
## 3 visual benefit cost\_… PC VR 32 32 -0.656 31 0.517  
## 4 audio cost cost\_… PC VR 32 32 0.174 31 0.863  
## 5 audio.visual cost cost\_… PC VR 32 32 -0.664 31 0.512  
## 6 visual cost cost\_… PC VR 32 32 -0.0545 31 0.957  
## # … with abbreviated variable name ¹​statistic

# group the data by both condition and cost benefit. and perform a paried t-test, on the cost benefit value.  
# corrected with bonferoni.  
# this was done to chek for in condition diffrences between the stimuly type  
ATTENTION %>%  
 filter(!is.na(cost\_ben\_val)) %>%  
 group\_by(cost\_benefit, condition) %>%  
 t\_test(cost\_ben\_val ~ stimuli\_type ,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 12 × 12  
## condition cost\_b…¹ .y. group1 group2 n1 n2 statis…² df p p.adj  
## \* <chr> <chr> <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl> <dbl>  
## 1 PC benefit cost… audio audio… 32 32 -1.42 31 0.167 0.501  
## 2 PC benefit cost… audio visual 32 32 0.293 31 0.771 1   
## 3 PC benefit cost… audio… visual 32 32 1.24 31 0.225 0.675  
## 4 VR benefit cost… audio audio… 32 32 -2.15 31 0.04 0.119  
## 5 VR benefit cost… audio visual 32 32 -0.931 31 0.359 1   
## 6 VR benefit cost… audio… visual 32 32 1.09 31 0.286 0.858  
## 7 PC cost cost… audio audio… 32 32 1.19 31 0.242 0.726  
## 8 PC cost cost… audio visual 32 32 0.00918 31 0.993 1   
## 9 PC cost cost… audio… visual 32 32 -1.07 31 0.291 0.873  
## 10 VR cost cost… audio audio… 32 32 0.212 31 0.834 1   
## 11 VR cost cost… audio visual 32 32 -0.214 31 0.832 1   
## 12 VR cost cost… audio… visual 32 32 -0.335 31 0.74 1   
## # … with 1 more variable: p.adj.signif <chr>, and abbreviated variable names  
## # ¹​cost\_benefit, ²​statistic

# Additionally, we calculate the means and standard deviations of cost benefit results for each combination of "cost\_benefit" and "stimuli\_type".  
ATTENTION %>%  
 group\_by(cost\_benefit, stimuli\_type, condition) %>%  
 get\_summary\_stats(cost\_ben\_val, type = "mean\_sd")

## # A tibble: 12 × 7  
## stimuli\_type condition cost\_benefit variable n mean sd  
## <chr> <chr> <chr> <chr> <dbl> <dbl> <dbl>  
## 1 audio PC benefit cost\_ben\_val 32 114. 222.  
## 2 audio VR benefit cost\_ben\_val 32 84.4 272.  
## 3 audio.visual PC benefit cost\_ben\_val 32 206. 330.  
## 4 audio.visual VR benefit cost\_ben\_val 32 222. 302.  
## 5 visual PC benefit cost\_ben\_val 32 94.8 358.  
## 6 visual VR benefit cost\_ben\_val 32 148. 295.  
## 7 audio PC cost cost\_ben\_val 32 -48.8 261.  
## 8 audio VR cost cost\_ben\_val 32 -58.6 189.  
## 9 audio.visual PC cost cost\_ben\_val 32 -155. 345.  
## 10 audio.visual VR cost cost\_ben\_val 32 -77.3 452.  
## 11 visual PC cost cost\_ben\_val 32 -49.6 422.  
## 12 visual VR cost cost\_ben\_val 32 -44.4 315.

#  
ATTENTION %>%  
 group\_by(stimuli\_type, condition) %>%  
 get\_summary\_stats(RT, type = "mean\_sd")

## # A tibble: 6 × 6  
## stimuli\_type condition variable n mean sd  
## <chr> <chr> <chr> <dbl> <dbl> <dbl>  
## 1 audio PC RT 96 1609. 328.  
## 2 audio VR RT 96 1934. 364.  
## 3 audio.visual PC RT 96 1713. 403.  
## 4 audio.visual VR RT 96 1957. 414.  
## 5 visual PC RT 96 1713. 389.  
## 6 visual VR RT 96 1943. 412.

ATTENTION %>%  
 group\_by(stimuli\_compatibility, condition) %>%  
 get\_summary\_stats(corrects, type = "mean\_sd")

## # A tibble: 6 × 6  
## stimuli\_compatibility condition variable n mean sd  
## <chr> <chr> <chr> <dbl> <dbl> <dbl>  
## 1 congruent PC corrects 96 12.9 2.54  
## 2 congruent VR corrects 96 11.8 2.54  
## 3 incongurent PC corrects 96 10.0 3.88  
## 4 incongurent VR corrects 96 9.95 3.12  
## 5 neutral PC corrects 96 12.0 2.69  
## 6 neutral VR corrects 96 11.0 2.65

ATTENTION %>%  
 group\_by(stimuli\_type, condition) %>%  
 get\_summary\_stats(corrects, type = "mean\_sd")

## # A tibble: 6 × 6  
## stimuli\_type condition variable n mean sd  
## <chr> <chr> <chr> <dbl> <dbl> <dbl>  
## 1 audio PC corrects 96 12.8 2.39  
## 2 audio VR corrects 96 12.1 2.28  
## 3 audio.visual PC corrects 96 11.0 3.65  
## 4 audio.visual VR corrects 96 10.4 3.01  
## 5 visual PC corrects 96 11.1 3.47  
## 6 visual VR corrects 96 10.2 2.93

ATTENTION %>%  
 group\_by(stimuli\_compatibility, condition) %>%  
 get\_summary\_stats(RT, type = "mean\_sd")

## # A tibble: 6 × 6  
## stimuli\_compatibility condition variable n mean sd  
## <chr> <chr> <chr> <dbl> <dbl> <dbl>  
## 1 congruent PC RT 96 1604. 332.  
## 2 congruent VR RT 96 1874. 356.  
## 3 incongurent PC RT 96 1678. 407.  
## 4 incongurent VR RT 96 1945. 419.  
## 5 neutral PC RT 96 1753. 377.  
## 6 neutral VR RT 96 2015. 403.

ATTENTION %>%  
 group\_by(stimuli\_type, stimuli\_compatibility, condition) %>%  
 get\_summary\_stats(corrects, type = "mean\_sd")

## # A tibble: 18 × 7  
## stimuli\_type stimuli\_compatibility condition variable n mean sd  
## <chr> <chr> <chr> <chr> <dbl> <dbl> <dbl>  
## 1 audio congruent PC corrects 32 12.9 2.08  
## 2 audio congruent VR corrects 32 12.3 2.27  
## 3 audio incongurent PC corrects 32 12.6 2.66  
## 4 audio incongurent VR corrects 32 11.8 2.44  
## 5 audio neutral PC corrects 32 12.8 2.46  
## 6 audio neutral VR corrects 32 12.2 2.17  
## 7 audio.visual congruent PC corrects 32 13.0 2.83  
## 8 audio.visual congruent VR corrects 32 11.8 2.35  
## 9 audio.visual incongurent PC corrects 32 8.5 3.91  
## 10 audio.visual incongurent VR corrects 32 8.88 3.19  
## 11 audio.visual neutral PC corrects 32 11.6 2.52  
## 12 audio.visual neutral VR corrects 32 10.6 2.76  
## 13 visual congruent PC corrects 32 12.7 2.71  
## 14 visual congruent VR corrects 32 11.3 2.95  
## 15 visual incongurent PC corrects 32 8.94 3.64  
## 16 visual incongurent VR corrects 32 9.22 2.94  
## 17 visual neutral PC corrects 32 11.6 2.97  
## 18 visual neutral VR corrects 32 10.1 2.58

ATTENTION %>%  
 group\_by(stimuli\_type, stimuli\_compatibility, condition) %>%  
 get\_summary\_stats(RT, type = "mean\_sd")

## # A tibble: 18 × 7  
## stimuli\_type stimuli\_compatibility condition variable n mean sd  
## <chr> <chr> <chr> <chr> <dbl> <dbl> <dbl>  
## 1 audio congruent PC RT 32 1566. 317.  
## 2 audio congruent VR RT 32 1881. 316.  
## 3 audio incongurent PC RT 32 1611. 312.  
## 4 audio incongurent VR RT 32 1927. 398.  
## 5 audio neutral PC RT 32 1651. 360.  
## 6 audio neutral VR RT 32 1995. 376.  
## 7 audio.visual congruent PC RT 32 1602. 336.  
## 8 audio.visual congruent VR RT 32 1860. 378.  
## 9 audio.visual incongurent PC RT 32 1702. 423.  
## 10 audio.visual incongurent VR RT 32 1956. 421.  
## 11 audio.visual neutral PC RT 32 1835. 423.  
## 12 audio.visual neutral VR RT 32 2056. 431.  
## 13 visual congruent PC RT 32 1644. 348.  
## 14 visual congruent VR RT 32 1881. 381.  
## 15 visual incongurent PC RT 32 1721. 474.  
## 16 visual incongurent VR RT 32 1953. 449.  
## 17 visual neutral PC RT 32 1775. 330.  
## 18 visual neutral VR RT 32 1994. 409.

##  
  
  
  
## Stimuli type X compatibility differences  
ATTENTION %>%  
 group\_by(stimuli\_type, stimuli\_compatibility) %>%  
 t\_test(corrects ~ condition ,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 9 × 10  
## stimuli\_type stimuli\_co…¹ .y. group1 group2 n1 n2 stati…² df p  
## \* <chr> <chr> <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl>  
## 1 audio congruent corr… PC VR 32 32 0.950 31 0.349   
## 2 audio incongurent corr… PC VR 32 32 1.67 31 0.104   
## 3 audio neutral corr… PC VR 32 32 0.897 31 0.377   
## 4 audio.visual congruent corr… PC VR 32 32 2.17 31 0.0379  
## 5 audio.visual incongurent corr… PC VR 32 32 -0.552 31 0.585   
## 6 audio.visual neutral corr… PC VR 32 32 1.43 31 0.162   
## 7 visual congruent corr… PC VR 32 32 1.95 31 0.0601  
## 8 visual incongurent corr… PC VR 32 32 -0.464 31 0.646   
## 9 visual neutral corr… PC VR 32 32 2.59 31 0.0146  
## # … with abbreviated variable names ¹​stimuli\_compatibility, ²​statistic

ATTENTION %>%  
 group\_by(stimuli\_type, stimuli\_compatibility) %>%  
 t\_test(RT ~ condition , p.adjust.method = "bonferroni", paired = T)

## # A tibble: 9 × 10  
## stimuli\_type stimuli\_c…¹ .y. group1 group2 n1 n2 stati…² df p  
## \* <chr> <chr> <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl>  
## 1 audio congruent RT PC VR 32 32 -5.54 31 4.53e-6  
## 2 audio incongurent RT PC VR 32 32 -4.66 31 5.69e-5  
## 3 audio neutral RT PC VR 32 32 -4.76 31 4.29e-5  
## 4 audio.visual congruent RT PC VR 32 32 -4.34 31 1.42e-4  
## 5 audio.visual incongurent RT PC VR 32 32 -3.14 31 3.69e-3  
## 6 audio.visual neutral RT PC VR 32 32 -2.46 31 1.97e-2  
## 7 visual congruent RT PC VR 32 32 -3.87 31 5.27e-4  
## 8 visual incongurent RT PC VR 32 32 -2.33 31 2.65e-2  
## 9 visual neutral RT PC VR 32 32 -3.15 31 3.56e-3  
## # … with abbreviated variable names ¹​stimuli\_compatibility, ²​statistic

## either Stimuli type or compatibility differences  
ATTENTION %>%  
 group\_by(stimuli\_type) %>%  
 t\_test(corrects ~ condition,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 3 × 9  
## stimuli\_type .y. group1 group2 n1 n2 statistic df p  
## \* <chr> <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl>  
## 1 audio corrects PC VR 96 96 2.02 95 0.0462  
## 2 audio.visual corrects PC VR 96 96 1.62 95 0.109   
## 3 visual corrects PC VR 96 96 2.30 95 0.0234

ATTENTION %>%  
 group\_by(stimuli\_compatibility) %>%  
 t\_test(corrects ~ condition,  
 p.adjust.method = "bonferroni",  
 paired = T)

## # A tibble: 3 × 9  
## stimuli\_compatibility .y. group1 group2 n1 n2 stati…¹ df p  
## \* <chr> <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl>  
## 1 congruent corrects PC VR 96 96 2.96 95 0.00387  
## 2 incongurent corrects PC VR 96 96 0.208 95 0.836   
## 3 neutral corrects PC VR 96 96 2.81 95 0.0061   
## # … with abbreviated variable name ¹​statistic

ATTENTION %>%  
 group\_by(stimuli\_type) %>%  
 t\_test(RT ~ condition , p.adjust.method = "bonferroni", paired = T)

## # A tibble: 3 × 9  
## stimuli\_type .y. group1 group2 n1 n2 statistic df p  
## \* <chr> <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl>  
## 1 audio RT PC VR 96 96 -8.62 95 1.47e-13  
## 2 audio.visual RT PC VR 96 96 -5.50 95 3.24e- 7  
## 3 visual RT PC VR 96 96 -5.11 95 1.63e- 6

ATTENTION %>%  
 group\_by(stimuli\_compatibility) %>%  
 t\_test(RT ~ condition, p.adjust.method = "bonferroni", paired = T)

## # A tibble: 3 × 9  
## stimuli\_compatibility .y. group1 group2 n1 n2 statistic df p  
## \* <chr> <chr> <chr> <chr> <int> <int> <dbl> <dbl> <dbl>  
## 1 congruent RT PC VR 96 96 -7.94 95 4.01e-12  
## 2 incongurent RT PC VR 96 96 -5.57 95 2.37e- 7  
## 3 neutral RT PC VR 96 96 -5.84 95 7.42e- 8

#

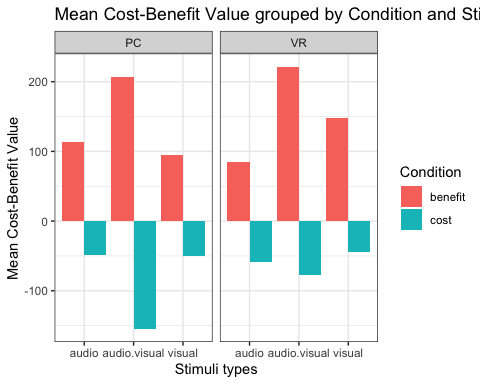
# The Response Time Coefficient of Variability (RTCV) is a measure of the variability in response times for a given task or stimulus. It is calculated as the ratio of the standard deviation of response times to the mean response time, expressed as a percentage.  
# The formula for calculating RTCV is:  
# RTCV = (standard deviation of response times / mean response time) x 100%  
# A higher RTCV indicates that response times are more variable or inconsistent, while a lower RTCV indicates more consistent or predictable response times.  
  
RTCV\_table <- ATTENTION %>%  
 group\_by(condition, stimuli\_type) %>%  
 get\_summary\_stats(RT, type = "mean\_sd") %>%  
 mutate(RTCV = sd / mean \* 100)  
  
RTCV\_table\_total <- ATTENTION %>%  
 group\_by(condition) %>%  
 get\_summary\_stats(RT, type = "mean\_sd") %>%  
 mutate(RTCV = sd / mean \* 100)  
  
RTCV\_table

## # A tibble: 6 × 7  
## stimuli\_type condition variable n mean sd RTCV  
## <chr> <chr> <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 audio PC RT 96 1609. 328. 20.4  
## 2 audio.visual PC RT 96 1713. 403. 23.5  
## 3 visual PC RT 96 1713. 389. 22.7  
## 4 audio VR RT 96 1934. 364. 18.8  
## 5 audio.visual VR RT 96 1957. 414. 21.2  
## 6 visual VR RT 96 1943. 412. 21.2

RTCV\_table\_total

## # A tibble: 2 × 6  
## condition variable n mean sd RTCV  
## <chr> <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 PC RT 288 1679. 377. 22.4  
## 2 VR RT 288 1945. 396. 20.4

# summary\_tavble represent the means and sd for the conditions and stimuli types, makes it easier to plot similar to matuz  
summary\_table <- ATTENTION %>%  
 group\_by(cost\_benefit, stimuli\_type, condition) %>%  
 get\_summary\_stats(cost\_ben\_val, type = "mean\_sd")  
# plotting the cost and benefit diffrences.  
ggplot(summary\_table, aes(x = stimuli\_type , y = mean, fill = cost\_benefit)) +  
 geom\_bar(stat = "identity", position = "dodge") +  
 theme\_bw() +  
 facet\_wrap( ~ condition) +  
 labs(title = "Mean Cost-Benefit Value grouped by Condition and Stimuli Type",  
 x = "Stimuli types",  
 y = "Mean Cost-Benefit Value",  
 fill = "Condition")



# plotting all the permutations for each condition.  
ggplot(ATTENTION,  
 aes(x = stimuli\_type, y = RT, fill = stimuli\_compatibility)) +  
 geom\_violin(trim = FALSE,  
 alpha = 0.5,  
 position = "dodge") +  
 geom\_jitter(  
 alpha = 0.9,  
 size = 1,  
 shape = 21,  
 width = 1  
 ) +  
 facet\_wrap( ~ condition) +  
 labs(x = "Stimuli Type",  
 y = "Search Time",  
 fill = "Stimuli Compatibility",  
 color = "Stimuli Compatibility") +  
 theme\_bw() +  
 scale\_fill\_manual(values = c("#F8766D", "#00BFC4", "black")) +  
 scale\_color\_manual(values = c("#F8766D", "#00BFC4", "black"))

